THE RANDOM FEATURE MODEL FOR INPUT-OUTPUT MAPS BETWEEN BANACH SPACES*

NICHOLAS H. NELSEN† AND ANDREW M. STUART‡

Abstract. Well known to the machine learning community, the random feature model, originally introduced by Rahimi and Recht in 2008, is a parametric approximation to kernel interpolation or regression methods. It is typically used to approximate functions mapping a finite-dimensional input space to the real line. In this paper, we instead propose a methodology for use of the random feature model as a data-driven surrogate for operators that map an input Banach space to an output Banach space. Although the methodology is quite general, we consider operators defined by partial differential equations (PDEs); here, the inputs and outputs are themselves functions, with the input parameters being functions required to specify the problem, such as initial data or coefficients, and the outputs being solutions of the problem. Upon discretization, the model inherits several desirable attributes from this infinite-dimensional, function space viewpoint, including mesh-invariant approximation error with respect to the true PDE solution map and the capability to be trained at one mesh resolution and then deployed at different mesh resolutions. We view the random feature model as a non-intrusive data-driven emulator, provide a mathematical framework for its interpretation, and demonstrate its ability to efficiently and accurately approximate the nonlinear parameter-to-solution maps of two prototypical PDEs arising in physical science and engineering applications: viscous Burgers' equation and a variable coefficient elliptic equation.

Key words. random feature, surrogate model, emulator, parametric PDE, solution map, high-dimensional approximation, model reduction, supervised learning, data-driven scientific computing

AMS subject classifications. 65D15, 65D40, 62M45, 35R60

1. Introduction.

The goal of this paper is to frame the *random feature model*, introduced in [57], as a methodology for the data-driven approximation of maps between infinite-dimensional spaces. Canonical examples of such maps include the semigroup generated by a time-dependent partial differential equation (PDE) mapping the initial condition (an input parameter) to the solution at a later time and the operator mapping a coefficient function (an input parameter) appearing in a PDE to its solution. Obtaining efficient and potentially low-dimensional representations of PDE solution maps is not only conceptually interesting, but also practically useful. Many applications in science and engineering require repeated evaluations of a complex and expensive forward model for different configurations of a system parameter. The model often represents a discretized PDE and the parameter, serving as input to the model, often represents a high-dimensional discretized quantity such as an initial condition or uncertain coefficient field. These *outer loop* applications commonly arise in inverse problems or uncertainty quantification tasks that involve control, optimization, or inference [56]. Full order forward models do not perform well in such many-query contexts, either due to excessive computational cost (requiring the most powerful high performance computing architectures) or slow evaluation time (unacceptable in real-time contexts such as on-the-fly optimal control). In contrast to the

*Submitted to the editors May 20, 2020.

Funding: NHN is supported by the National Science Foundation Graduate Research Fellowship Program under award DGE-1745301. AMS is supported by NSF (award DMS-1818977) and by the Office of Naval Research (award N00014-17-1-2079). Both authors are supported by NSF (award AGS-1835860).

†Mechanical and Civil Engineering, California Institute of Technology, Pasadena, CA 91125, USA (nnelsen@caltech.edu).

‡Computing and Mathematical Sciences, California Institute of Technology, Pasadena, CA 91125, USA (astuart@caltech.edu).

big data regime that dominates computer vision and other technological fields, only a relatively small amount of high resolution data is generated from computer simulations or physical experiments in scientific applications. Fast approximate solvers built from this limited available data that can efficiently and accurately emulate the full order model would be highly advantageous.

In this work, we demonstrate that the random feature model holds considerable potential for such a purpose. Resembling [50] and the contemporaneous work in [11, 48], we present a methodology for true function space learning of black-box input-output maps between a Banach space and separable Hilbert space. We formulate the approximation problem as supervised learning in infinite dimensions and show that the natural hypothesis space is a reproducing kernel Hilbert space associated with an operator-valued kernel. For a suitable loss functional, training the random feature model is equivalent to solving a finite-dimensional convex optimization problem. As a consequence of our careful construction of the method as mapping between Banach spaces, the resulting emulator naturally scales favorably with respect to the high input and output dimensions arising in practical, discretized applications; furthermore, it is shown to achieve small relative test error for two model problems arising from approximation of a semigroup and the solution map for an elliptic PDE exhibiting parametric dependence on a coefficient function.

1.1. Literature Review. In recent years, two different lines of research have emerged that address PDE approximation problems with machine learning techniques. The first perspective takes a more traditional approach akin to point collocation methods from the field of numerical analysis. Here, the goal is to use a deep neural network (NN) to solve a prescribed initial boundary value problem with as high accuracy as possible. Given a point cloud in a spatio-temporal domain $\tilde{D}$ as input data, the prevailing approach first directly parametrizes the PDE solution field as a NN and then optimizes the NN parameters by minimizing the PDE residual with respect to (w.r.t.) some loss functional (see [58, 64, 71] and the references therein). To clarify, the object approximated with this novel method is a *low-dimensional* input-output map $\tilde{D} \to \mathbb{R}$, i.e., the real-valued function that solves the PDE. This approach is mesh-free by definition but highly intrusive as it requires full knowledge of the specified PDE. Any change to the original formulation of the initial boundary value problem or related PDE problem parameters necessitates an (expensive) re-training of the NN solution. We do not explore this first approach any further in this article.

The second direction is arguably more ambitious: use a NN as an emulator for the infinite-dimensional mapping between an input parameter and the PDE solution itself or a functional of the solution, i.e., a quantity of interest; the latter is widely prevalent in uncertainty quantification problems. We emphasize that the object approximated in this setting, unlike in the aforementioned first approach, is an input-output map $\mathcal{X} \to \mathcal{Y}$, i.e., the PDE solution operator, where $\mathcal{X}$, $\mathcal{Y}$ are infinite-dimensional Banach spaces; this map is generally nonlinear. For an approximation-theoretic treatment of parametric PDEs in general, we refer the reader to the article of Cohen and DeVore [19]. In applications, the solution operator is represented by a discretized forward model $\mathbb{R}^K \to \mathbb{R}^K$, where K is the mesh size, and hence represents a *high-dimensional* object. It is this second line of research that inspires our work.

Of course, there are many approaches to forward model reduction that do not explicitly involve machine learning ideas. The reduced basis method (see [4, 7, 24] and the references therein) is a classical idea based on constructing an empirical basis from data snapshots and solving a cheaper variational problem; it is still widely used

in practice due to computationally efficient offline-online decompositions that eliminate dependence on the full order degrees of freedom. Recently, machine learning extensions to the reduced basis methodology, of both intrusive (e.g., projection-based reduced order models) and non-intrusive (e.g., model-free data only) type, have further improved the applicability of these methods [17, 29, 37, 46, 62]. However, the input-output maps considered in these works involve high dimension in only one of the input or output space, not both. Other popular surrogate modeling techniques include Gaussian processes [74], polynomial chaos expansions [65], and radial basis functions [72]; yet, these are only practically suitable for problems with input space of low to moderate dimension. Classical numerical methods for PDEs may also represent the forward model $\mathbb{R}^K \to \mathbb{R}^K$, albeit implicitly in the form a computer code (e.g.: finite element, finite difference, finite volume methods). However, the approximation error is sensitive to K and repeated evaluations of this forward model often becomes cost prohibitive due to poor scaling with input dimension K.

Instead, deep NNs have been identified as strong candidate surrogate models for parametric PDE problems due to their empirical ability to emulate high-dimensional nonlinear functions with minimal evaluation cost once trained. Early work in the use of NNs to learn the solution operator, or vector field, defining ODEs and time-dependent PDEs, may be found in the 1990s [33, 59]. There are now more theoretical justifications for NNs breaking the *curse of dimensionality* [45, 53], leading to increased interest in PDE applications [30, 63]. A suite of work on data-driven discretizations of PDEs has emerged that allow for identification of the governing model [3, 12, 49, 67]; we note that only the operators appearing in the equation itself are approximated with these approaches, not the solution operator of the PDE. More in line with our focus in this article, architectures based on deep convolutional NNs have proven quite successful for learning elliptic PDE solution maps (for example, see [68, 75, 76], which take an image-to-image regression approach). Other NNs have been used in similar elliptic problems for quantity of interest prediction [43], error estimation [15], or unsupervised learning [47]. Yet in all the approaches above, the architectures and resulting error are dependent on the mesh resolution. To circumvent this issue, the surrogate map must be well-defined on function space and independent of any finite-dimensional realization of the map that arises from discretization. This is not a new idea (see [16, 60] or for functional data analysis, [40, 54]). The aforementioned reduced basis method is an example, as is the method of [18, 19], which approximates the solution map with sparse Taylor polynomials and is proved to achieve optimal convergence rates in idealized settings. However, it is only recently that machine learning methods have been explicitly designed to operate in an infinite-dimensional setting, and there is little work in this direction [11, 48]. Here we propose the random feature model as another such method.

The random feature model (RFM) [57], detailed in Subsection 2.3, is in some sense the simplest possible machine learning model; it may be viewed as an ensemble average of randomly parametrized functions: an expansion in a randomized basis. These *random features* could be defined, for example, by randomizing the internal parameters of a NN. Compared to NN emulators with enormous learnable parameter counts (e.g., $\mathcal{O}(10^5)$ to $\mathcal{O}(10^6)$, see [27, 28, 47]) and methods that are intrusive or lead to nontrivial implementations [18, 46, 62], the RFM is one of the simplest models to formulate and train (often $\mathcal{O}(10^3)$ parameters, or fewer, suffice). The theory of the RFM for real-valued outputs is well developed, partly due to its close connection to kernel methods [13, 39, 57] and Gaussian processes [73], and includes generalization rates and dimension-free estimates [53, 57, 66]. A quadrature viewpoint on the RFM

provides further insight and leads to Monte Carlo sampling ideas [2]; we explore this further in Subsection 2.3. As in modern deep learning practice, the RFM has also been shown to perform best when the model is over-parametrized [6]. In a similar high-dimensional setting of relevance in this paper, the authors of [34, 42] theoretically investigated nonparametric kernel regression for parametric PDEs with real-valued solution map outputs. However, these works require explicit knowledge of the kernel itself, rather than working with random features that implicitly define a kernel as we do here; furthermore, our work considers both infinite-dimensional input *and* output spaces, not just one or the other. A key idea underlying our approach is to formulate the proposed random feature algorithm on infinite-dimensional space and only then discretize. This philosophy in algorithm development has been instructive in a number of areas in scientific computing, such as optimization [38] and the development of Monte Carlo Markov Chain methodology [21]. It has recently been promoted as a way of designing and analyzing algorithms within machine learning [35, 52, 61, 69, 70] and our work may be understood within this general framework.

1.2. Contributions. Our primary contributions in this paper are now listed.

1. We develop the random feature model, directly formulated on the function space level, for learning input-output maps between Banach spaces purely from data. As a method for parametric PDEs, the methodology is non-intrusive but also has the additional advantage that it may be used in settings where only data is available and no model is known.
2. We show that our proposed method is more computationally tractable to both train and evaluate than standard kernel methods in infinite dimensions. Furthermore, we show that the method is equivalent to kernel ridge regression performed in a finite-dimensional space spanned by random features.
3. We apply our methodology to learn the semigroup defined by the solution operator for viscous Burgers' equation and the coefficient-to-solution operator for the Darcy flow equation.
4. We demonstrate, by means of numerical experiments, two mesh-independent approximation properties that are built into the proposed methodology: invariance of relative error to mesh resolution and evaluation ability on any mesh resolution.

This paper is structured as follows. In Section 2, we communicate the mathematical framework required to work with the random feature model in infinite dimensions, identify an appropriate approximation space, and explain the training procedure. We introduce two instantiations of random feature maps that target physical science applications in Section 3 and detail the corresponding numerical results for these applications in Section 4. We conclude in Section 5 with discussion and future work.

2. Methodology. In this work, the overarching problem of interest is the approximation of a map $F^\dagger : \mathcal{X} \to \mathcal{Y}$, where $\mathcal{X}$, $\mathcal{Y}$ are infinite-dimensional spaces of real-valued functions defined on some bounded open subset of $\mathbb{R}^d$ and $F^\dagger$ is defined by $a \mapsto F^\dagger(a) := u$, where u is the solution of a (possibly time dependent) PDE and a is an input function required to make the problem well-posed. Our proposed approach for this approximation, constructing a surrogate map F for the true map $F^\dagger$, is data-driven, non-intrusive, and based on least squares. Least squares-based methods are integral to the random feature methodology as proposed in low dimensions [57] and generalized here to the infinite-dimensional setting; they have also been shown to work well in other algorithms for high-dimensional numerical approximation [10, 20, 25]. Within the broader scope of reduced order modeling techniques [7], the approach we

adopt in this paper falls within the class of data-fit emulators. In its essence, our method interpolates the solution manifold

$$\mathcal{M} = \{u \in \mathcal{Y} : u = F^\dagger(a),\, a \in \mathcal{X}\}\,. \tag{2.1}$$

The solution map $F^\dagger$, as the inverse of a differential operator, is often smoothing and admits a notion of compactness, i.e., the output space compactly embeds into the input space. Then, the idea is that $\mathcal{M}$ should have some compact, low-dimensional structure (intrinsic dimension). However, actually finding a model F that exploits this structure despite the high dimensionality of the truth map $F^\dagger$ is quite difficult. Further, the effectiveness of many model reduction techniques, such as those based on the reduced basis method, are dependent on inherent properties of the map $F^\dagger$ itself (e.g., analyticity), which in turn may influence the decay rate of the Kolmogorov n-width of the manifold $\mathcal{M}$ [19]. While such subtleties of approximation theory are crucial to developing rigorous theory and provably convergent algorithms, we choose to work in the non-intrusive setting where knowledge of the map $F^\dagger$ and its associated PDE are only obtained through measurement data, and hence detailed characterizations such as those mentioned above are essentially unavailable.

The remainder of this section introduces the mathematical preliminaries for our methodology. With the goal of operator approximation in mind, in Subsection 2.1 we formulate a supervised learning problem in an infinite-dimensional setting. We provide the necessary background on reproducing kernel Hilbert spaces in Subsection 2.2 and then define the RFM in Subsection 2.3. In Subsection 2.4, we describe the optimization principle which leads to algorithms for the RFM and an example problem in which $\mathcal{X}$ and $\mathcal{Y}$ are one-dimensional spaces.

2.1. Problem Formulation. Let $\mathcal{X}$, $\mathcal{Y}$ be real Banach spaces and $F^\dagger : \mathcal{X} \to \mathcal{Y}$ a (possibly nonlinear) map. It is natural to frame the approximation of $F^\dagger$ as a supervised learning problem. Suppose we are given training data in the form of input-output pairs $\{a_i, y_i\}_{i=1}^n \subset \mathcal{X} \times \mathcal{Y}$, where $a_i \sim \nu$ i. i. d., ν is a probability measure supported on $\mathcal{X}$, and $y_i = F^\dagger(a_i) \sim F^\dagger_\sharp \nu$ with, potentially, noise added to the evaluations of $F^\dagger(\cdot)$; in our applications, this noise may be viewed as resulting from model error (the PDE does not perfectly represent the physics) and/or from discretization error (in approximating the PDE). We aim to build a parametric reconstruction of the true map $F^\dagger$ from the data, that is, construct a model $F : \mathcal{X} \times \mathcal{P} \to \mathcal{Y}$ and find $\alpha^\dagger \in \mathcal{P} \subseteq \mathbb{R}^m$ such that $F(\cdot, \alpha^\dagger) \approx F^\dagger$ are close as maps from $\mathcal{X}$ to $\mathcal{Y}$ in some suitable sense. The natural number m here denotes the total number of parameters in the model. The standard approach to determine parameters in supervised learning is to first define a loss functional $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$ and then minimize the expected risk,

$$\min_{\alpha \in \mathcal{P}} \mathbb{E}^{a\sim\nu}[\ell(F^\dagger(a), F(a, \alpha))]\,. \tag{2.2}$$

With only the data $\{a_i, y_i\}_{i=1}^n$ at our disposal, we approximate problem (2.2) by replacing ν with the empirical measure $\nu^{(n)} = \frac{1}{n}\sum_{j=1}^n \delta_{a_j}$, which leads to the empirical risk minimization problem

$$\min_{\alpha \in \mathcal{P}} \frac{1}{n}\sum_{j=1}^n \ell(y_j, F(a_j, \alpha))\,. \tag{2.3}$$

The hope is that given minimizer $\alpha^{(m)}$ of (2.3) and $\alpha^\dagger$ of (2.2), $F(\cdot, \alpha^{(m)})$ well approximates $F(\cdot, \alpha^\dagger)$, that is, the learned model *generalizes* well; these ideas may be

made rigorous with results from statistical learning theory [36]. Solving problem (2.3) is called *training* the model F. Once trained, the model is then validated on a new set of i. i. d. input-output pairs previously unseen during the training process. This *testing* phase indicates how well F approximates $F^\dagger$. From here on out, we assume that $(\mathcal{Y}, \langle\cdot,\cdot\rangle_{\mathcal{Y}}, \|\cdot\|_{\mathcal{Y}})$ is a real separable Hilbert space and focus on the squared loss

$$\ell(y_1, y_2) := \frac{1}{2}\|y_1 - y_2\|_{\mathcal{Y}}^2 . \tag{2.4}$$

We stress that our entire formulation is in an infinite-dimensional setting and that we will remain in this setting throughout the paper; as such, the random feature methodology we propose will inherit desirable discretization-invariant properties, to be observed in the numerical experiments of Section 4.

2.2. Operator-Valued Reproducing Kernels. The random feature model is naturally formulated in a reproducing kernel Hilbert space (RKHS) setting, as our exposition will show in Subsection 2.3. However, the usual RKHS theory is concerned with real-valued functions [1, 8, 22, 72]. Our setting, with the output space $\mathcal{Y}$ a separable Hilbert space, requires several new ideas that generalize the real-valued case. We now outline these ideas; parts of the presentation that follow may be found in the references [2, 14, 54].

We first consider the special case $\mathcal{Y} := \mathbb{R}$ for ease of exposition. A real RKHS is a Hilbert space $(\mathcal{H}, \langle\cdot,\cdot\rangle_{\mathcal{H}}, \|\cdot\|_{\mathcal{H}})$ comprised of real-valued functions $f : \mathcal{X} \to \mathbb{R}$ such that the pointwise evaluation functional $f \mapsto f(a)$ is bounded for every $a \in \mathcal{X}$. It then follows that there exists a unique, symmetric, positive definite kernel function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ such that for every $a \in \mathcal{X}$, $k(\cdot, a) \in \mathcal{H}$ and the *reproducing kernel property* $f(a) = \langle k(\cdot, a), f\rangle_{\mathcal{H}}$ holds. These two properties are often taken as the definition of an RKHS. The converse direction is also true: every symmetric, positive definite kernel defines a unique RKHS.

We now introduce the needed generalization of the reproducing property to arbitrary real Hilbert spaces $\mathcal{Y}$, as this result will motivate the construction of the random feature model. With elements of $\mathcal{Y}$ now arbitrary elements of a vector space, the kernel is now operator-valued.

Definition 2.1. *Let $\mathcal{X}$ be a real Banach space and $\mathcal{Y}$ a real separable Hilbert space. An* **operator-valued kernel** *is a map*

$$k : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{Y}, \mathcal{Y}) , \tag{2.5}$$

where $\mathcal{L}(\mathcal{Y}, \mathcal{Y})$ denotes the Banach space of all bounded linear operators on $\mathcal{Y}$, such that its adjoint satisfies $k(a, a')^* = k(a', a)$ for all a, $a' \in \mathcal{X}$ and for every $N \in \mathbb{N}$,

$$\sum_{i,j=1}^{N} \langle y_i, k(a_i, a_j) y_j\rangle_{\mathcal{Y}} \geq 0 \tag{2.6}$$

for all pairs $\{(a_i, y_i)\}_{i=1}^N \subset \mathcal{X} \times \mathcal{Y}$.

Paralleling the development for the real-valued case, an operator-valued kernel k also uniquely (up to isomorphism) determines an associated real RKHS $\mathcal{H}_k = \mathcal{H}_k(\mathcal{X}; \mathcal{Y})$. Now, choosing a probability measure ν supported on $\mathcal{X}$, we define a kernel integral operator (in the sense of the Bochner integral) by

$$\begin{aligned} T_k : L^2_\nu(\mathcal{X}; \mathcal{Y}) &\to L^2_\nu(\mathcal{X}; \mathcal{Y}) \\ F &\mapsto T_k F := \int k(\cdot, a') F(a') \nu(da') , \end{aligned} \tag{2.7}$$

which is non-negative, self-adjoint, and compact (provided $k(a,a) \in \mathcal{L}(\mathcal{Y},\mathcal{Y})$ is compact for all $a \in \mathcal{X}$ [14]). Let us further assume that all conditions needed for $T_k^{1/2}$ to be an isometric isomorphism from L^2_ν into $\mathcal{H}_k$ are satisfied. Generalizing the standard Mercer theory (see, e.g., [2, 8]), we may write the RKHS inner product using

$$\langle F, G\rangle_{\mathcal{H}_k} = \left\langle F, T_k^{-1} G\right\rangle_{L^2_\nu} \quad \forall F, G \in \mathcal{H}_k\,. \tag{2.8}$$

Note that while (2.8) appears to depend on the measure ν on $\mathcal{X}$, the RKHS $\mathcal{H}_k$ is itself determined by the kernel without any reference to a measure (see [22], Chp. 3, Thm. 4). With the inner product now explicit, we may directly deduce a reproducing property. A fully rigorous justification of the methodology is outside the scope of this article; however, we perform formal computations which provide intuition underpinning the methodology. To this end we fix $a \in \mathcal{X}$ and $y \in \mathcal{Y}$. Then,

$$\begin{aligned}
\left\langle k(\cdot,a)y, T_k^{-1}F\right\rangle_{L^2_\nu} &= \int \left\langle k(a',a)y, (T_k^{-1}F)(a')\right\rangle_{\mathcal{Y}} \nu(da') \\
&= \int \left\langle y, k(a,a')(T_k^{-1}F)(a')\right\rangle_{\mathcal{Y}} \nu(da') \\
&= \left\langle y, \int k(a,a')(T_k^{-1}F)(a')\,\nu(da')\right\rangle_{\mathcal{Y}} \\
&= \langle y, F(a)\rangle_{\mathcal{Y}}\,,
\end{aligned}$$

by using Definition 2.1 of operator-valued kernel and the fact that $k(\cdot,a)y \in \mathcal{H}_k$ ([14]). So, we deduce the following:

Result 2.2 (Reproducing property for operator-valued kernels). *Let $F \in \mathcal{H}_k$ be given. Then for every $a \in \mathcal{X}$ and $y \in \mathcal{Y}$,*

$$\langle y, F(a)\rangle_{\mathcal{Y}} = \langle k(\cdot,a)y, F\rangle_{\mathcal{H}_k}\,. \tag{2.9}$$

This identity, paired with a special choice of k, is the basis of the random feature model in our abstract infinite-dimensional setting.

2.3. Random Feature Model. One could approach the approximation of $F^\dagger : \mathcal{X} \to \mathcal{Y}$ from the perspective of kernel methods. However, it is generally a difficult task to explicitly design operator-valued kernels of the form (2.5) since the spaces $\mathcal{X}$, $\mathcal{Y}$ may be of different regularity, for example. Example constructions of operator-valued kernels studied in the literature include diagonal operators, multiplication operators, and composition operators [40, 54], but these all involve some simple generalization of scalar-valued kernels. Instead, the random feature model allows one to implicitly work with operator-valued kernels by choosing a *random feature map* $\varphi : \mathcal{X} \times \Theta \to \mathcal{Y}$ and a probability measure μ supported on Θ; the map φ is assumed to be square integrable w.r.t. the product measure $\nu \times \mu$. We now show the connection between random features and kernels; to this end, recall the following standard notation:

Notation 2.3. Given $(H, \langle\cdot,\cdot\rangle, \|\cdot\|)$ a Hilbert space, the *outer product* $a \otimes b \in \mathcal{L}(H,H)$ is defined by $(a\otimes b)c = \langle b, c\rangle a$ for any $a, b, c \in H$. ◊

Then, we consider maps $k_\mu : \mathcal{X}\times\mathcal{X} \to \mathcal{L}(\mathcal{Y},\mathcal{Y})$ of the form

$$k_\mu(a,a') := \int \varphi(a;\theta)\otimes\varphi(a';\theta)\mu(d\theta)\,. \tag{2.10}$$

Since k_μ may readily be shown to be an operator-valued kernel via Definition 2.1, it defines a unique real RKHS $\mathcal{H}_{k_\mu} \subset L^2_\nu(\mathcal{X};\mathcal{Y})$. Our approximation theory will be based on this space or finite-dimensional approximations thereof.

We now perform a purely formal but instructive calculation, following from application of the reproducing property (2.9) to operator-valued kernels of the form (2.10). Doing so leads to an integral representation of any $F^\dagger \in \mathcal{H}_{k_\mu}$: for all $a \in \mathcal{X}$, $y \in \mathcal{Y}$,

$$\begin{aligned}\big\langle y, F^\dagger(a)\big\rangle_{\mathcal{Y}} = \big\langle k_\mu(\cdot,a)y, F^\dagger\big\rangle_{\mathcal{H}_{k_\mu}} &= \left\langle \int \langle \varphi(a;\theta), y\rangle_{\mathcal{Y}}\, \varphi(\cdot;\theta)\, \mu(d\theta), F^\dagger \right\rangle_{\mathcal{H}_{k_\mu}} \\ &= \int \langle \varphi(a;\theta), y\rangle_{\mathcal{Y}} \big\langle \varphi(\cdot;\theta), F^\dagger\big\rangle_{\mathcal{H}_{k_\mu}} \mu(d\theta) \\ &= \int c(\theta) \langle y, \varphi(a;\theta)\rangle_{\mathcal{Y}}\, \mu(d\theta) \\ &= \left\langle y, \int c(\theta)\varphi(a;\theta)\mu(d\theta) \right\rangle_{\mathcal{Y}},\end{aligned}$$

where the coefficient function $c : \Theta \to \mathbb{R}$ is defined by

$$c(\theta) := \big\langle \varphi(\cdot;\theta), F^\dagger\big\rangle_{\mathcal{H}_{k_\mu}}. \tag{2.11}$$

Since $\mathcal{Y}$ is Hilbert, the above holding for all $y \in \mathcal{Y}$ implies the integral representation

$$F^\dagger = \int c(\theta)\varphi(\cdot;\theta)\mu(d\theta)\,. \tag{2.12}$$

The expression for $c(\theta)$ needs careful interpretation because $\varphi(\cdot;\theta) \notin \mathcal{H}_{k_\mu}$ with probability one; indeed, $c(\theta)$ is defined only as an L^2_μ limit. Nonetheless, the RKHS may be completely characterized by this integral representation. Define

$$\begin{aligned}\mathcal{A} : L^2_\mu(\Theta;\mathbb{R}) &\to L^2_\nu(\mathcal{X};\mathcal{Y}) \\ c &\mapsto \mathcal{A}c := \int c(\theta)\varphi(\cdot;\theta)\mu(d\theta)\,.\end{aligned} \tag{2.13}$$

Then we have the following result whose proof, provided in Appendix A, is a straightforward generalization of the real-valued case given in [2], Sec. 2.2:

Result 2.4. *Under the assumption that $\varphi \in L^2_{\nu\times\mu}(\mathcal{X}\times\Theta;\mathcal{Y})$, the RKHS defined by the kernel k_μ in (2.10) is precisely*

$$\mathcal{H}_{k_\mu} = \mathrm{im}(\mathcal{A}) = \left\{ \int c(\theta)\varphi(\cdot;\theta)\mu(d\theta) : c \in L^2_\mu(\Theta;\mathbb{R}) \right\}. \tag{2.14}$$

We stress that the integral representation (2.12) is not unique since $\mathcal{A}$ is not injective in general. A central role in what follows is the approximation of measure μ by the empirical measure

$$\mu^{(m)} := \frac{1}{m}\sum_{j=1}^m \delta_{\theta_j}\,, \quad \theta_j \sim \mu \ \text{ i. i. d.} \tag{2.15}$$

Given this, define $k^{(m)} := k_{\mu^{(m)}}$ to be the empirical approximation to k_μ:

$$k^{(m)}(a,a') = \mathbb{E}^{\theta\sim\mu^{(m)}}[\varphi(a;\theta)\otimes\varphi(a';\theta)] = \frac{1}{m}\sum_{j=1}^m \varphi(a;\theta_j)\otimes\varphi(a';\theta_j)\,. \tag{2.16}$$

Then define $\mathcal{H}_{k^{(m)}}$ to be the unique RKHS induced by the kernel $k^{(m)}$. The following characterization of $\mathcal{H}_{k^{(m)}}$ is proved in Appendix A:

RESULT 2.5. *Assume that $\varphi \in L^2_{\nu\times\mu}(\mathcal{X}\times\Theta;\mathcal{Y})$ and that the random features $\{\varphi(\cdot;\theta_j)\}_{j=1}^m$ are linearly independent in $L^2_\nu(\mathcal{X};\mathcal{Y})$. Then, the RKHS $\mathcal{H}_{k^{(m)}}$ is equal to the linear span of the $\{\varphi_j := \varphi(\cdot;\theta_j)\}_{j=1}^m$.*

Applying a simple Monte Carlo sampling approach to Equation (2.12), specifically, replacing the probability measure μ by the empirical measure $\mu^{(m)}$, gives the approximation

$$F^\dagger \approx \frac{1}{m}\sum_{j=1}^m c(\theta_j)\varphi(\cdot;\theta_j)\,; \tag{2.17}$$

by virtue of Result 2.5, this approximation is in $\mathcal{H}_{k^{(m)}}$ and achieves the Monte Carlo rate $\mathcal{O}(m^{-1/2})$. However, in the setting of interest to us, the Monte Carlo approach does not give rise to a practical method for two reasons: evaluation of $c(\theta_j)$ requires knowledge of both the unknown mapping $F^\dagger$ and of the RKHS appearing in the inner product defining c from $F^\dagger$; in our setting the kernel k_μ of the RKHS is not assumed to be known – only the random features are assumed to be given. To sidestep these difficulties, the RFM adopts a data-driven optimization approach to determine a different approximation to $F^\dagger$, also from the space $\mathcal{H}_{k^{(m)}}$.

We now define the RFM:

DEFINITION 2.6. *Given probability space $(\mathcal{X},\nu)$, with $\mathcal{X}$ a real Banach space, probability space (Θ,μ), with Θ a finite or infinite-dimensional Banach space, real separable Hilbert space $\mathcal{Y}$, and $\varphi \in L^2_{\nu\times\mu}(\mathcal{X}\times\Theta;\mathcal{Y})$, the* **random feature model** *is the parametric map*

$$\begin{aligned} &F_m : \mathcal{X}\times\mathbb{R}^m \to \mathcal{Y} \\ &\qquad (a;\alpha) \mapsto F_m(a;\alpha) := \frac{1}{m}\sum_{j=1}^m \alpha_j\varphi(a;\theta_j)\,, \quad \theta_j \sim \mu \text{ i.i.d.} \end{aligned} \tag{2.18}$$

We implicitly use the Borel σ-algebra to define the probability spaces in the preceding definition. The goal of the RFM is to choose parameters $\alpha \in \mathbb{R}^m$ so as to approximate mappings $F^\dagger \in \mathcal{H}_{k_\mu}$ by mappings $F_m(\cdot;\alpha) \in \mathcal{H}_{k^{(m)}}$. The RFM may be viewed as a *spectral method* since the randomized basis $\varphi(\cdot;\theta)$ in the linear expansion (2.18) is defined on all of $\mathcal{X}$. Determining the coefficient vector α from data obviates the difficulties associated with the Monte Carlo approach since the method only requires knowledge of sample input-output pairs from $F^\dagger$ and knowledge of the random feature map φ.

As written, Equation (2.18) is incredibly simple. It is clear that the choice of random feature map and measure pair (φ,μ) will determine the quality of approximation. In their original paper [57], Rahimi and Recht took a kernel-oriented perspective by first choosing a kernel and then finding a random feature map to estimate this kernel. Our perspective is the opposite in that we allow the choice of random feature map φ to implicitly *define* the kernel via the formula (2.10) instead of picking the kernel first. This methodology also has implications for numerics: the kernel never explicitly appears in any computations, which leads to storage savings. It does, however, leave open the question of characterizing the RKHS $\mathcal{H}_{k_\mu}$ of mappings from $\mathcal{X}$ to $\mathcal{Y}$ that underlies the approximation method.

The connection to kernels explains the origins of the RFM in the machine learning literature. Moreover, the RFM may also be interpreted in the context of neural networks. To see this, consider the setting where $\mathcal{X}$, $\mathcal{Y}$ are both equal to the Euclidean space $\mathbb{R}$ and choose φ to be a family of hidden neurons $\varphi_{NN}(a;\theta) := \sigma(\theta^{(1)} \cdot a + \theta^{(2)})$. A single hidden layer NN would seek to find $\{(\alpha_j, \theta_j)\}_{j=1}^m$ in $\mathbb{R} \times \mathbb{R}^2$ so that

$$\frac{1}{m} \sum_{j=1}^{m} \alpha_j \varphi_{NN}(a;\theta_j) \tag{2.19}$$

matches the given training data $\{a_i, y_i\}_{i=1}^n \subset \mathcal{X} \times \mathcal{Y}$. More generally, and in arbitrary Euclidean spaces, one may allow $\varphi_{NN}(\cdot;\theta)$ to be any deep NN. However, while the RFM has the same *form* as (2.19), there is a difference in the *training*: the θ_j are drawn i.i.d. from a probability measure and then fixed, and only the α_j are chosen to fit the training data. This connection is quite profound: given any deep NN with randomly initialized parameters θ, studies of the lazy training regime and neural tangent kernel [29, 39] suggest that adopting a RFM approach and optimizing over only α is quite natural, as it is observed that in this regime the NN parameters do not stray far from their random initialization during gradient descent whilst the last layer of parameters α_j adapt considerably.

Once the parameters $\{\theta_j\}_{j=1}^m$ are chosen at random and fixed, training the RFM only requires optimizing over $\alpha \in \mathbb{R}^m$ which, due to linearity of F_m in α, is a simple task to which we now turn our attention.

2.4. Optimization. One of the most attractive characteristics of the RFM is its training procedure. With the L^2-type loss (2.4) as in standard regression settings, optimizing the coefficients of the RFM with respect to the empirical risk (2.3) is a convex optimization problem, requiring only the solution of a finite-dimensional system of linear equations; the convexity also suggests the possibility of appending convex constraints (such as linear inequalities), although we do not pursue this here. We emphasize the simplicity of the underlying optimization tasks as they suggest the possibility of numerical implementation of the RFM into complicated black-box computer codes.

We now proceed to show that a regularized version of the optimization problem (2.3)–(2.4) arises naturally from approximation of a nonparametric regression problem defined over the RKHS $\mathcal{H}_{k_\mu}$. To this end, recall the supervised learning formulation in Subsection 2.1. Given n i. i. d. input-output pairs $\{a_i, y_i = F^\dagger(a_i)\}_{i=1}^n \subset \mathcal{X} \times \mathcal{Y}$ as data, with the a_i drawn from unknown probability measure ν on $\mathcal{X}$, the objective is to find an approximation F^* to the map $F^\dagger$. Let $\mathcal{H}_{k_\mu}$ be the hypothesis space and k_μ its operator-valued reproducing kernel of the form (2.10). The most straightforward learning algorithm in this RKHS setting is kernel ridge regression, also known as penalized least squares. This method produces a nonparametric model by finding a minimizer F^* of

$$\min_{F \in \mathcal{H}_{k_\mu}} \sum_{j=1}^{n} \frac{1}{2} \|y_j - F(a_j)\|_{\mathcal{Y}}^2 + \frac{\lambda}{2} \|F\|_{\mathcal{H}_{k_\mu}}^2 , \tag{2.20}$$

where $\lambda \geq 0$ is a penalty parameter. By the representer theorem for operator-valued kernels [54], the minimizer has the form

$$F^* = \sum_{\ell=1}^{n} k_\mu(\cdot, a_\ell)\beta_\ell \tag{2.21}$$

for some functions $\{\beta_j\}_{j=1}^n \subset \mathcal{Y}$. In practice, finding these n functions in the output space requires solving a (block) linear operator equation. For the high-dimensional PDE problems we consider in this work, solving such an equation may become prohibitively expensive from both operation count and storage required. A few workarounds were proposed in [40] such as certain diagonalizations, but these rely on simplifying assumptions that are quite limiting. More fundamentally, the representation of the solution in (2.21) requires knowledge of the kernel k_μ; in our setting we assume access only to the random features which define k_μ and not k_μ itself.

We thus proceed to explain how to make progress with this problem given only knowledge of random features. Recall the empirical kernel given by (2.16), the RKHS $\mathcal{H}_{k^{(m)}}$, and Result 2.5. The following result, proved in Appendix A, shows that a RFM hypothesis class with a penalized least squares empirical loss function in optimization problem (2.3)–(2.4) is equivalent to kernel ridge regression (2.20) restricted to $\mathcal{H}_{k^{(m)}}$.

RESULT 2.7. *Assume that $\varphi \in L^2_{\nu\times\mu}(\mathcal{X}\times\Theta;\mathcal{Y})$ and that the random features $\{\varphi(\cdot;\theta_j)\}_{j=1}^m$ are linearly independent in $L^2_\nu(\mathcal{X};\mathcal{Y})$. Fix $\lambda \geq 0$. Let $\alpha^* \in \mathbb{R}^m$ be the unique minimum norm solution of the following problem:*

$$\min_{\alpha\in\mathbb{R}^m} \sum_{j=1}^n \frac{1}{2}\left\| y_j - \frac{1}{m}\sum_{\ell=1}^m \alpha_\ell \varphi(a_j;\theta_\ell)\right\|_{\mathcal{Y}}^2 + \frac{\lambda}{2m}\|\alpha\|_2^2 . \tag{2.22}$$

Then, the RFM defined by this choice $\alpha = \alpha^*$ satisfies

$$F_m(\cdot;\alpha^*) = \operatorname*{argmin}_{F\in\mathcal{H}_{k^{(m)}}} \sum_{j=1}^n \frac{1}{2}\|y_j - F(a_j)\|_{\mathcal{Y}}^2 + \frac{\lambda}{2}\|F\|_{\mathcal{H}_{k^{(m)}}}^2 . \tag{2.23}$$

Solving the convex problem (2.22) trains the RFM. The first order condition for a global minimizer leads to the normal equations

$$\frac{1}{m}\sum_{i=1}^m\sum_{j=1}^n \alpha_i \langle \varphi(a_j;\theta_i), \varphi(a_j;\theta_\ell)\rangle_{\mathcal{Y}} + \lambda\alpha_\ell = \sum_{j=1}^n \langle y_j, \varphi(a_j;\theta_\ell)\rangle_{\mathcal{Y}} \tag{2.24}$$

for each $\ell \in \{1,\dots,m\}$. This is an m-by-m linear system of equations for $\alpha \in \mathbb{R}^m$ that is standard to solve. In the case $\lambda = 0$, the minimum norm solution may be written in terms of a pseudoinverse operator [51].

Example 2.8 (Brownian bridge). We now provide a simple one-dimensional instantiation of the random feature model to illustrate the methodology. Denote the input by $a = x$ and take the input space $\mathcal{X} := (0,1)$, output space $\mathcal{Y} := \mathbb{R}$, input space measure $\nu(dx) := dx$, and random parameter space $\Theta := \ell^\infty(\mathbb{N};\mathbb{R})$. Then, consider the random feature map $\varphi : (0,1)\times\ell^\infty(\mathbb{N};\mathbb{R}) \to \mathbb{R}$ defined by the *Brownian bridge*

$$\varphi(x;\theta) := \sum_{j=1}^\infty \theta^{(j)}(j\pi)^{-1}\sqrt{2}\sin(j\pi x), \quad \theta^{(j)} \sim N(0,1) \text{ i.i.d.}, \tag{2.25}$$

where $\theta = \{\theta^{(j)}\}_{j\in\mathbb{N}}$. For any realization of θ, the function $\varphi(\cdot;\theta)$ is a Brownian motion constrained to zero at $x = 0$ and $x = 1$. The kernel $k : (0,1)\times(0,1) \to \mathbb{R}$ is simply the covariance function of this stochastic process:

$$k(x,x') = \mathbb{E}^{\theta^{(j)}\sim N(0,1)}[\varphi(x;\theta)\varphi(x';\theta)] = \min\{x,x'\} - xx' . \tag{2.26}$$

Note that k is the Green's function for the negative Laplacian on $(0,1)$ with Dirichlet boundary conditions. Using this fact, we may explicitly characterize the associated RKHS $\mathcal{H}_k$ as follows. First, define

$$T_k f := \int_0^1 k(\cdot, y) f(y)\, dy = \left(-\frac{d^2}{dx^2}\right)^{-1} f\,, \tag{2.27}$$

where the the negative Laplacian has domain $H_0^1((0,1);\mathbb{R}) \cap H^2((0,1);\mathbb{R})$. Viewing T_k as an operator from $L^2((0,1);\mathbb{R})$ into itself, from (2.8) we conclude, upon integration by parts,

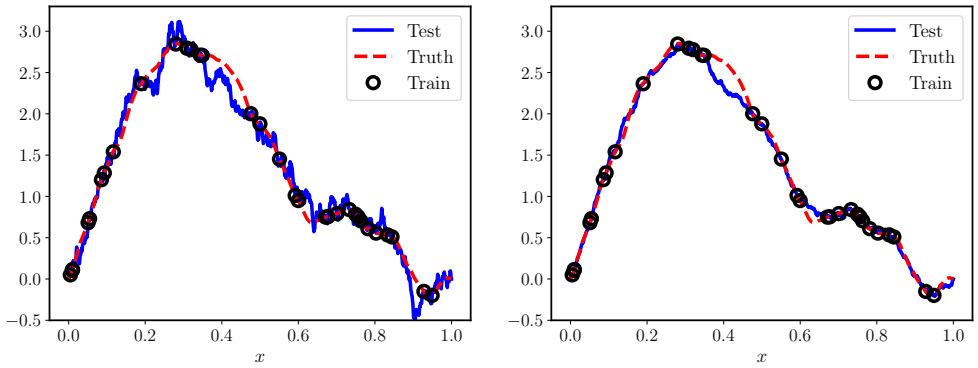
$$\langle f, g\rangle_{\mathcal{H}_k} = \langle f, T_k^{-1} g\rangle_{L^2} = \left\langle \frac{df}{dx}, \frac{dg}{dx}\right\rangle_{L^2} = \langle f, g\rangle_{H_0^1} \quad \forall f, g \in \mathcal{H}_k\,; \tag{2.28}$$

note that the last identity does indeed define an inner product on H_0^1. By this formal argument we identify the RKHS $\mathcal{H}_k$ as the Sobolev space $H_0^1((0,1);\mathbb{R})$. Furthermore, Brownian bridge may be viewed as the Gaussian measure $N(0, T_k)$. Approximation using the RFM with the Brownian bridge random feature map is illustrated in Figure 1. Since $k(\cdot, x)$ is a piecewise linear function, a kernel regression method will produce a piecewise linear approximation. Indeed, the figure indicates that the RFM with n training points fixed approaches the optimal piecewise linear approximation as $m \to \infty$ (see [53] for a related theoretical result). ♢

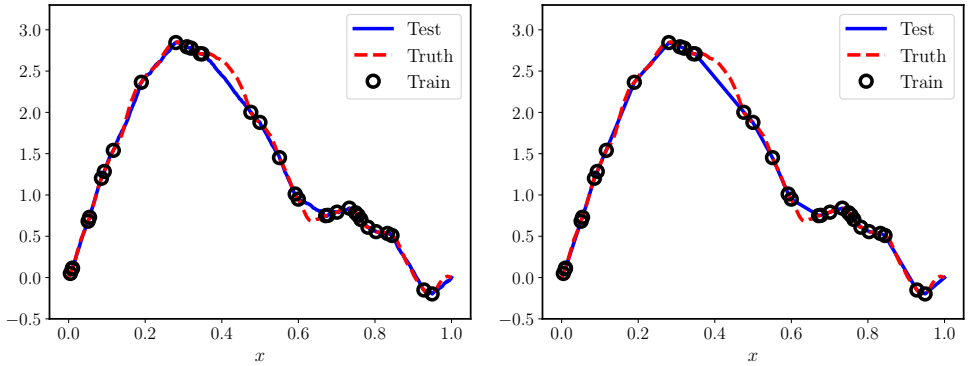
The Brownian bridge example illuminates a more fundamental idea. For this low-dimensional problem, an expansion in a deterministic Fourier sine basis would of course be natural. But if we do not have a natural, computable orthonormal basis, then randomness provides a useful alternative representation; notice that the random features each include random combinations of the deterministic Fourier sine basis in this Brownian bridge example. For the more complex problems that we move on to study numerically in the next two sections, we lack knowledge of good, computable bases for general maps in infinite dimensions. The RFM approach exploits randomness to allow us to explore, implicitly discover the structure of, and represent, such maps. Thus we now turn away from this example of real-valued maps defined on a subset of the real line and instead consider the use of random features to represent maps between spaces of functions.

3. Application to PDE Solution Maps. In this section, we design the random feature maps $\varphi : \mathcal{X} \times \Theta \to \mathcal{Y}$ for the RFM approximation of two particular PDE parameter-to-solution maps: the evolution semigroup of viscous Burgers' equation in Subsection 3.1 and the coefficient-to-solution operator for the Darcy problem in Subsection 3.2. As practitioners of kernel methods in machine learning have found, the choice of kernel (which in this work, follows from the choice of random feature map) plays a central role in the quality of the function reconstruction. While our method is purely data-driven and requires no knowledge of the governing PDE, we take the view that any prior knowledge can, and should, be introduced into the design of φ. The maps φ that we employ are nonlinear in both arguments. We also detail the probability measure ν placed on the input space $\mathcal{X}$ for each of the two PDE applications; this choice is crucial because while it is desirable that the trained RFM generalizes to arbitrary inputs in $\mathcal{X}$, we can in general only expect to learn an approximation of the truth map $F^\dagger$ restricted to inputs that resemble those drawn from ν.

3.1. Burgers' Equation: Formulation. Viscous Burgers' equation in one spatial dimension is representative of the advection-dominated PDE problem class; these



(a) $m = 50$ (b) $m = 500$



(c) $m = 5000$ (d) $m = \infty$

Fig. 1: Brownian bridge random feature model for one-dimensional input-output spaces with $n = 32$ training points fixed and $\lambda = 0$ (Example 2.8): as $m \to \infty$, the RFM approaches the nonparametric regression solution given by the representer theorem (Figure 1d), which in this case is a piecewise linear approximation of the true function (an element of RKHS $\mathcal{H}_k = H_0^1$, shown in red). Blue lines denote the trained model evaluated on test data points and black circles denote training data points.

time-dependent equations are not conservation laws due to the presence of a small dissipative term, but nonlinear transport still plays a central role in the evolution of solutions. The initial value problem we consider is

$$\begin{cases} \dfrac{\partial u}{\partial t} + \dfrac{\partial}{\partial x}\left(\dfrac{u^2}{2}\right) - \varepsilon \dfrac{\partial^2 u}{\partial x^2} = f & \text{in } (0,\infty)\times(0,1) \\ u(\cdot,0) = u(\cdot,1)\,, \quad \dfrac{\partial u}{\partial x}(\cdot,0) = \dfrac{\partial u}{\partial x}(\cdot,1) & \text{in } (0,\infty) \\ u(0,\cdot) = a & \text{in } (0,1)\,, \end{cases} \tag{3.1}$$

where $\varepsilon > 0$ is the viscosity (i.e., diffusion coefficient) and we have imposed periodic boundary conditions. The initial condition a serves as the input and is drawn according to a Gaussian measure defined by

$$a \sim \nu := N(0, C)\,, \tag{3.2}$$

with covariance operator

$$C = \tau^{2\alpha-d}(-\Delta + \tau^2 \,\mathrm{Id})^{-\alpha}\,, \tag{3.3}$$

where $d = 1$ and the operator $-\Delta$ is defined on $\mathbb{T}^1$. The hyperparameter $\tau \in \mathbb{R}^+$ is an inverse length scale and $\alpha > 1/2$ controls the regularity of the draw. It is known that such a are Hölder with exponent up to $\alpha - 1/2$, so in particular $a \in \mathcal{X} := L^2((0,1);\mathbb{R})$. Then for all $\varepsilon > 0$, the unique global solution $u(t,\cdot)$ to (3.1) is real analytic for all $t > 0$ (see [44], Thm. 1.1). Hence, setting the output space to be $\mathcal{Y} := H^s((0,1);\mathbb{R})$ for any $s > 0$, we may define the solution map

$$\begin{aligned} F^\dagger &: L^2 \to H^s \\ &a \mapsto F^\dagger(a) := \Psi_T(a) = u(T,\cdot)\,, \end{aligned} \tag{3.4}$$

where $\{\Psi_t\}_{t>0}$ forms the solution operator semigroup for (3.1) and we fix the final time $t = T > 0$. The map $F^\dagger$ is smoothing and nonlinear.

We now describe a random feature map for use in the RFM (2.18) that we call *Fourier space random features*. Let $\mathcal{F}$ denote the Fourier transform and define $\varphi : \mathcal{X} \times \Theta \to \mathcal{Y}$ by

$$\varphi(a;\theta) = \sigma(\mathcal{F}^{-1}(g\mathcal{F}\theta\mathcal{F}a))\,, \tag{3.5}$$

where $\sigma(\cdot)$, the ELU function defined below, is defined as a mapping on $\mathbb{R}$ and applied pointwise to vectors. The randomness enters through $\theta \sim \mu := N(0, C')$, with C' the same covariance operator as in (3.3) but with potentially different inverse length scale and regularity, and the *wavenumber filter function* $g : \mathbb{R} \to \mathbb{R}^+$ is

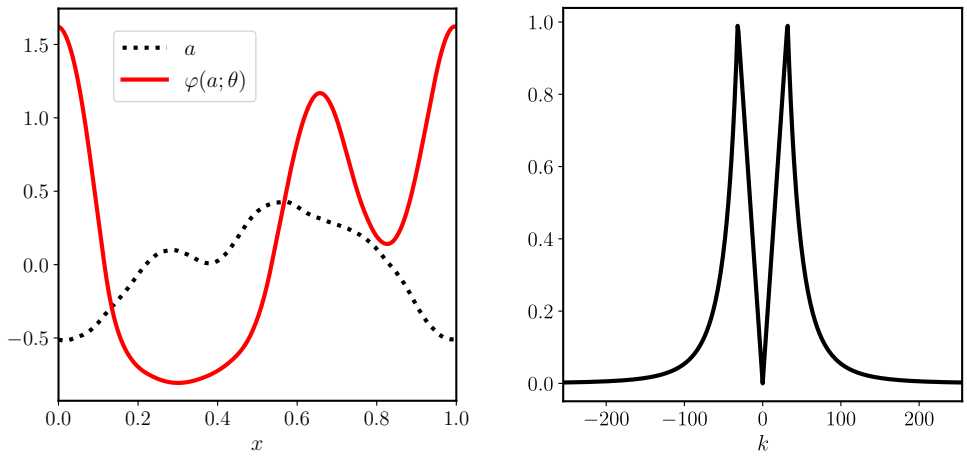
$$g(k) = \sigma_g(\delta|k|)\,, \quad \sigma_g(r) := \max\{0, \min\{2r, (r+1/2)^{-\beta}\}\}\,, \tag{3.6}$$

where $\delta,\, \beta > 0$. The map $\varphi(\cdot;\theta)$ essentially performs a random convolution with the initial condition followed by a filtering operation. Figure 2a illustrates a sample input and output from φ. While not optimized for performance, the filter g is designed to shuffle the energy in low to medium wavenumbers and cut off high wavenumbers (see Figure 2b), reflecting our prior knowledge of the behavior of solutions to (3.1).

We choose the activation function σ in (3.5) to be the exponential linear unit $\mathrm{ELU} : \mathbb{R} \to \mathbb{R}$ defined by

$$\mathrm{ELU}(r) := \begin{cases} r\,, & r \geq 0 \\ e^r - 1\,, & r < 0\,. \end{cases} \tag{3.7}$$

ELU has successfully been used as activation in other machine learning frameworks for related nonlinear PDE problems [46, 55]. We also find ELU to perform better in the RFM framework over several other choices including $\mathrm{ReLU}(\cdot)$, $\tanh(\cdot)$, $\mathrm{sigmoid}(\cdot)$, $\sin(\cdot)$, $\mathrm{SELU}(\cdot)$, and $\mathrm{softplus}(\cdot)$. Note that the pointwise evaluation of ELU in (3.5) will be well defined, by Sobolev embedding, for $s > 1/2$ sufficiently large in the definition of $\mathcal{Y} = H^s$. Since the solution operator maps into H^s for any $s > 0$, this does not constrain the method.



(a) (b)

Fig. 2: Random feature map construction for Burgers' equation: Figure 2a displays a representative input-output pair for the random feature $\varphi(\cdot;\theta)$ (Equation (3.5)) while Figure 2b shows the filter function $g(k)$ for $\delta = 0.0025$ and $\beta = 4$ (Equation (3.6)).

3.2. Darcy Flow: Formulation. Divergence form elliptic equations [32] arise in a variety of applications, in particular, the groundwater flow in a porous medium governed by Darcy's law [5]. This linear elliptic boundary value problem reads

$$\begin{cases} -\nabla \cdot (a\nabla u) = f & \text{in } D \\ u = 0 & \text{on } \partial D\,, \end{cases} \tag{3.8}$$

where D is a bounded open subset in $\mathbb{R}^d$, f represents sources and sinks of fluid, a the permeability of the porous medium, and u the piezometric head; all three functions map D into $\mathbb{R}$ and, in addition, a is strictly positive almost everywhere in D. We work in a setting where f is fixed and consider the input-output map defined by $a \mapsto u$. The measure ν on a is a high contrast level set prior constructed as a pushforward of a Gaussian measure:

$$a \sim \nu := \psi_\sharp N(0, C)\,; \tag{3.9}$$

here $\psi : \mathbb{R} \to \mathbb{R}$ is a threshold function defined by

$$\psi(r) = a^+ \mathbb{1}_{(0,\infty)}(r) + a^- \mathbb{1}_{(-\infty,0)}(r)\,, \quad 0 < a^- \le a^+ < \infty \tag{3.10}$$

and the covariance operator C is given in (3.3) with $d = 2$ and homogeneous Neumann boundary conditions on the Laplacian $-\Delta$. That is, the resulting coefficient a almost surely takes only two values (a^+ or a^-) and, as the zero level set of a Gaussian random field, exhibits random geometry in the physical domain D. It follows that $a \in L^\infty(D;\mathbb{R}^+)$ almost surely. Further, the size of the contrast ratio a^+/a^- measures the scale separation of this elliptic problem and hence controls the difficulty of reconstruction [9]. See Figure 3a for a representative draw.

Given $f \in L^2(D;\mathbb{R})$, the standard Lax-Milgram theory may be applied to show that for coefficient $a \in \mathcal{X} := L^\infty(D;\mathbb{R}^+)$, there exists a unique weak solution $u \in$

$\mathcal{Y} := H_0^1(D;\mathbb{R})$ for Equation (3.8) (see, e.g., Evans [26]). Thus, we define the ground truth solution map

$$\begin{aligned} F^\dagger : L^\infty &\to H_0^1 \\ a &\mapsto F^\dagger(a) := u\,. \end{aligned} \tag{3.11}$$

We emphasize that while the PDE (3.8) is linear, the solution map $F^\dagger$ is nonlinear.

We now describe the chosen random feature map for this problem, which we call *predictor-corrector random features*. Define $\varphi : \mathcal{X} \times \Theta \to \mathcal{Y}$ by $\varphi(a;\theta) := p_1$ such that

$$-\Delta p_0 = \frac{f}{a} + \sigma_\gamma(\theta_1) \tag{3.12a}$$

$$-\Delta p_1 = \frac{f}{a} + \sigma_\gamma(\theta_2) + \nabla(\log a)\cdot\nabla p_0\,, \tag{3.12b}$$

where the boundary conditions are homogeneous Dirichlet, $\theta = (\theta_1, \theta_2)$ are two Gaussian random fields each drawn from measure $N(0, C')$, f is the source term in (3.8), and $\gamma = (s^+, s^-, \delta)$ are parameters for a thresholded sigmoid $\sigma_\gamma : \mathbb{R} \to \mathbb{R}$ defined by

$$\sigma_\gamma(r) := \frac{s^+ - s^-}{1 + e^{-r/\delta}} + s^- \tag{3.13}$$

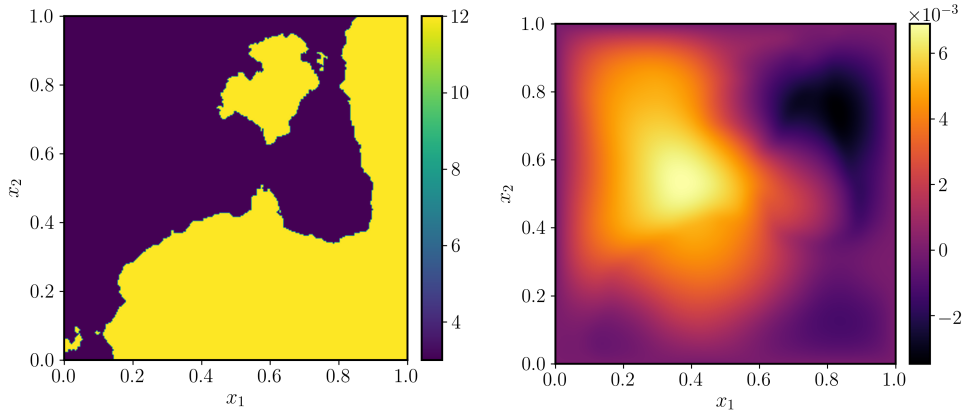
and extended as a Nemytskii operator when applied to $\theta_1(\cdot)$ and $\theta_2(\cdot)$. In practice, since ∇a is not well-defined when drawn from the level set measure, we replace a with a_ε, where $a_\varepsilon := v(1)$ is a smoothed version of a obtained by evolving the following linear heat equation

$$\begin{cases} \dfrac{dv}{dt} = \eta\Delta v & \text{in } (0,1)\times D \\ n\cdot\nabla v = 0 & \text{on } (0,1)\times \partial D \\ v(0) = a & \text{in } D \end{cases} \tag{3.14}$$

for one time unit. An example of the response of $\varphi(\cdot;\theta)$ to a piecewise constant input $a \sim \nu$ is shown in Figure 3.

We remark that by removing the two random terms involving θ_1, θ_2 in (3.12), we obtain a remarkably accurate surrogate model for the PDE. This observation is representative of a more general iterative method, a predictor-corrector type iteration, for solving the Darcy equation (3.8), which is convergent for sufficiently large coefficients a. The map φ is essentially a random perturbation of a single step of this iterative method: Equation (3.12a) makes a coarse prediction of the output, then (3.12b) improves this prediction with a correction term (here, this correction term is derived from expanding the original PDE). This choice of φ falls within an ensemble viewpoint that the RFM may be used to improve pre-existing surrogate models by taking $\varphi(\cdot;\theta)$ to be an existing emulator, but randomized in a principled way through θ.

We are cognizant of the fact that, for this particular example, the random feature map φ requires full knowledge of the Darcy equation and a naïve evaluation of φ may be as expensive as solving the original PDE, which is itself a linear PDE; however, we believe that the ideas underlying the random features used here are intuitive and suggestive of what is possible in other applications areas. For example, random feature models may be applied on domains with simple geometries, which are supersets of the physical domain of interest, enabling the use of fast tools such as the fast Fourier



(a) $a \sim \nu$ (b) $\varphi(a;\theta)\,,\ \theta \sim \mu$

Fig. 3: Random feature map construction for Darcy flow: Figure 3a displays a representative input draw a with $\tau = 3$, $\alpha = 2$ and $a^+ = 12$, $a^- = 3$; Figure 3b shows the output random feature $\varphi(a;\theta)$ (Equation (3.12)) taking the coefficient a as input. Here, $f \equiv 1$, $\tau' = 7.5$, $\alpha' = 2$, $s^+ = 1/a^+$, $s^- = -1/a^-$, and $\delta = 0.15$.

transform (FFT) within the RFM even though they may not be available on the original problem, either because the operator to be inverted is spatially inhomogeneous or because of the geometry of the physical domain.

4. Numerical Experiments. We now assess the performance of our proposed methodology on the approximation of operators $F^\dagger : \mathcal{X} \to \mathcal{Y}$ presented in Section 3. In the numerical experiments that follow, all infinite-dimensional objects are discretized on a uniform mesh with K degrees of freedom. In this section, our notation does not make explicit the dependence on K because the random feature model is consistent with the continuum in the limit $K \to \infty$; we demonstrate this fact numerically.

The input functions and our chosen random feature maps (3.5) and (3.12) require i. i. d. draws of Gaussian random fields to be fully defined. We efficiently sample these fields by truncating a Karhunen-Loéve expansion and employing fast summation of the eigenfunctions with FFT. More precisely, on a mesh of size K, denote by g_θ a numerical representation of a Gaussian random field on domain $D = (0,1)^d$, $d = 1, 2$:

$$g_\theta = \sum_{k \in Z_K} \xi_k \sqrt{\lambda_k}\phi_k \approx \sum_{k' \in (\mathbb{Z}^+)^d} \xi_{k'} \sqrt{\lambda_{k'}}\phi_{k'} \sim N(0, C)\,, \tag{4.1}$$

where $\xi_j \sim N(0,1)$ i. i. d. and $Z_K \subset \mathbb{Z}^+$ is a truncated one-dimensional lattice of cardinality K ordered such that λ_j is non-increasing. For clarity, the eigenvalue problem $C\phi_k = \lambda_k \phi_k$ for non-negative, symmetric, trace-class operator C in (3.3) has solutions

$$\phi_k(x) = 2\cos(k_1 \pi x_1)\cos(k_2 \pi x_2)\,, \quad \lambda_k = \tau^{2\alpha-2}(\pi^2|k|^2 + \tau^2)^{-\alpha} \tag{4.2}$$

for homogeneous Neumann boundary conditions when $d = 2$, $k = (k_1, k_2) \in (\mathbb{Z}^+)^2$, $x = (x_1, x_2) \in (0,1)^2$, and solutions

$$\phi_k(x) = e^{2\pi i k x}\,, \quad \lambda_k = \tau^{2\alpha-1}(4\pi^2 k^2 + \tau^2)^{-\alpha} \tag{4.3}$$

for periodic boundary conditions when $d = 1$, $k \in \mathbb{Z}$, $x \in (0, 1)$ (with appropriately modified random variables ξ_j in (4.1) to ensure that the resulting g_θ is real-valued). These forms of g_θ are used in all experiments that follow.

To measure the distance between the RFM approximation $F_m(\cdot; \alpha^*)$ and the ground truth map $F^\dagger$, we employ the *approximate expected relative test error*

$$e_{n',m} := \frac{1}{n'} \sum_{j=1}^{n'} \frac{\left\| F^\dagger(a_j) - F_m(a_j; \alpha^*) \right\|_{L^2}}{\left\| F^\dagger(a_j) \right\|_{L^2}} \approx \mathbb{E}^{a \sim \nu} \frac{\left\| F^\dagger(a) - F_m(a; \alpha^*) \right\|_{L^2}}{\left\| F^\dagger(a) \right\|_{L^2}}, \tag{4.4}$$

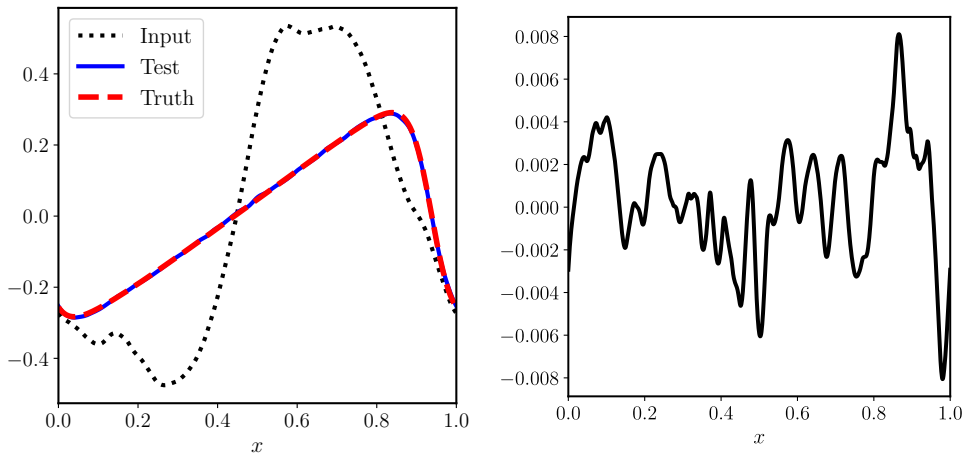
where the $\{a_j\}_{j=1}^{n'}$ are drawn i.i.d. from ν and n' denotes the number of input-output pairs used in testing. All $L^2(D; \mathbb{R})$ norms on the physical domain are numerically approximated by trapezoid rule quadrature. Since $\mathcal{Y} \subset L^2(D; \mathbb{R})$ for both the PDE solution operators (3.4) and (3.11), we also perform all required inner products during training in $L^2(D; \mathbb{R})$ rather than in $\mathcal{Y}$; this results in smaller relative test error $e_{n',m}$.

4.1. Burgers' Equation: Experiment. We generate a high resolution dataset of input-output pairs by solving Burgers' equation (3.1) on a uniform periodic mesh of size $K = 1025$ (identifying the first mesh point with the last) using an FFT-based pseudospectral method for spatial discretization and a fourth-order Runge-Kutta integrating factor time-stepping scheme [41] for time discretization. All mesh sizes $K < 1025$ are subsampled from this original dataset and hence we consider numerical realizations of $F^\dagger$ up to $\mathbb{R}^{1025} \to \mathbb{R}^{1025}$. We fix $n = 512$ training and $n' = 4000$ testing pairs unless otherwise noted. The input data are drawn from $\nu = N(0, C)$ where C is given by (3.3) with parameter choices $\tau = 7$ and $\alpha = 2.5$. We fix the viscosity to $\varepsilon = 10^{-2}$ in all experiments. Lowering ε leads to smaller length scale solutions and more difficult reconstruction; more data (higher n) and features (higher m) or a more expressive choice of φ would be required to achieve comparable error levels. For simplicity, we set the forcing $f \equiv 0$, although nonzero forcing could lead to other interesting solution maps such as $f \mapsto u(T, \cdot)$. It is easy to check that the solution will have zero mean for all time and a steady state of zero. Hence, we choose $T \leq 2$ to ensure that the solution is far from attaining steady state. For the random feature map (3.5), we fix the hyperparameters $\alpha' = 2$, $\tau' = 5$, $\delta = 0.0025$, and $\beta = 4$. The map itself is evaluated efficiently with the FFT, and hyperparameters were not optimized. We find that regularization during training has a negligible effect for our problem, so the RFM is trained with $\lambda = 0$ by solving the normal equations (2.24) with the pseudoinverse to deliver the minimum norm least squares solution; we use the truncated SVD implementation in `scipy.linalg.pinv2` for this purpose.

Our experiments study the RFM approximation to the viscous Burgers' equation evolution operator semigroup (3.4). As a visual aid for the high-dimensional problem at hand, Figure 4 shows a representative sample input and output along with a trained RFM test prediction. To determine whether the RFM has actually learned the correct evolution operator, we test the semigroup property of the map. Denote the $(j-1)$-fold composition of a function F with itself by F^j. Then, with $u(0, \cdot) = a$, we have

$$(\Psi_T \circ \cdots \circ \Psi_T)(a) = \Psi_T^j(a) = \Psi_{jT}(a) = u(rT, \cdot) \tag{4.5}$$

by definition. We train the RFM on input-output pairs from the map Ψ_T with $T := 0.5$ to obtain $F_* := F_m(\cdot; \alpha^*)$. Then, it should follow from (4.5) that $F_*^j \approx \Psi_{jT}$, that is, each application of F_* should evolve the solution T time units. We test this semigroup approximation by learning the map F_* and then comparing F_*^j on $n' = 4000$ fixed



(a) (b)

Fig. 4: Representative input-output test sample for the Burgers' equation solution map $F^\dagger := \Psi_1$: Here, $n = 512$, $m = 1024$, and $K = 1025$. Figure 4a shows a sample input, output (truth), and trained RFM prediction (test), while Figure 4b displays the pointwise error. The relative L^2 error for this single prediction is 0.0146.

inputs to outputs from each of the operators Ψ_{jT}, with $j \in \{1, 2, 3, 4\}$ (the solutions at time T, $2T$, $3T$, $4T$). The results are presented in Table 1 for a fixed mesh size $K = 129$. We observe that the composed RFM map F_*^j accurately captures Ψ_{jT},

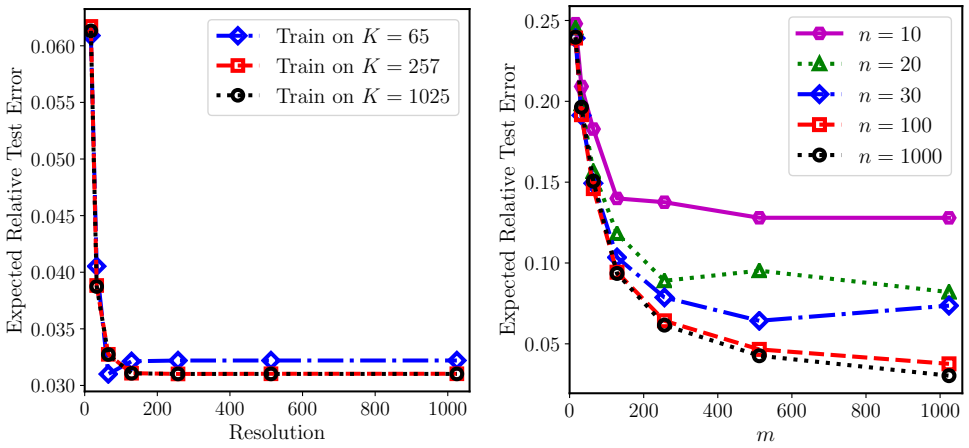
Train on:	$T = 0.5$	Test on:	$2T = 1.0$	$3T = 1.5$	$4T = 2.0$
	0.0360		0.0407	0.0528	0.0788

Table 1: Expected relative error $e_{n',m}$ for time upscaling with the learned RFM operator semigroup for Burgers' equation: Here, $n' = 4000$, $m = 1024$, $n = 512$, and $K = 129$. The RFM is trained on data from the evolution operator $\Psi_{T=0.5}$, and then tested on input-output samples generated from Ψ_{jT}, where $j = 2$, 3, 4, by repeated composition of the learned model. The error increase is small even after three compositions of the learned RFM map, reflecting excellent generalization ability.

though this accuracy deteriorates as j increases due to error propagation. However, even after three compositions corresponding to 1.5 time units past the training time $T = 0.5$, the relative error only increases by around 0.04. It is remarkable that the RFM learns time evolution without explicitly time-stepping the PDE (3.1) itself. Such a procedure is coined *time upscaling* in the PDE context and in some sense breaks the CFL stability barrier [23]. Table 1 is evidence that the RFM has excellent generalization properties: while only trained on inputs $a \sim \nu$, the model predicts well on new input samples $\Psi_{jT}(a) \sim (\Psi_{jT})_\sharp \nu$.

We next study the ability of the RFM to transfer its learned coefficients α^* obtained from training on mesh size K to different mesh resolutions K' in Figure 5a. We fix $T := 1$ from here on and observe that the lowest test error occurs when $K = K'$,

that is, when the train and test resolutions are identical; this behavior was also observed in the contemporaneous work [48]. Still, the errors are essentially constant across resolution, indicating that the RFM learns its optimal coefficients independently of the resolution and hence generalizes well to any desired mesh size. In fact, the trained model could employ different discretization methodologies (e.g.: finite element, spectral) for its random feature map, not just different mesh sizes.

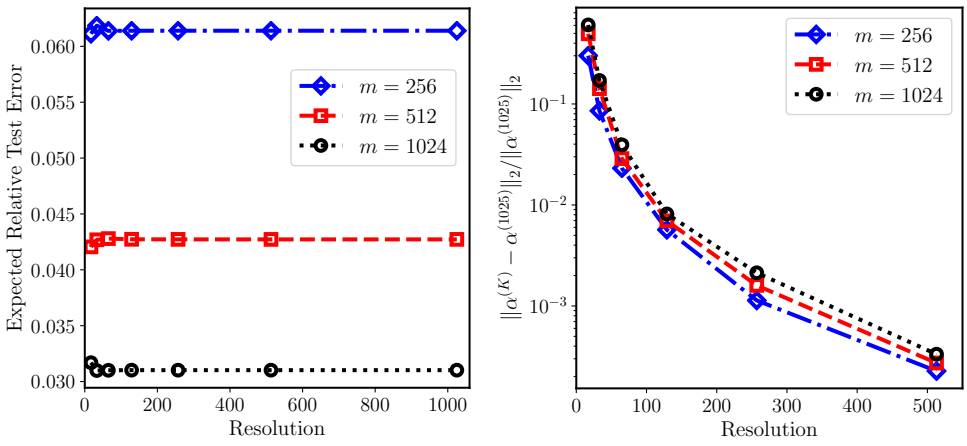


(a) (b)

Fig. 5: Expected relative test error of a trained RFM for the Burgers' equation evolution operator $F^\dagger = \Psi_1$ with $n' = 4000$ test pairs: Figure 5a displays the invariance of test error w.r.t. training and testing on different resolutions for $m = 1024$ and $n = 512$ fixed; the RFM can train and test on different mesh sizes without loss of accuracy. Figure 5b shows the decay of the test error for resolution $K = 129$ fixed as a function of m and n; the smallest error achieved is 0.0303 for $n = 1000$ and $m = 1024$.

The smallest expected relative test error achieved by the RFM is 0.0303 for the configuration detailed in Figure 5b. In this figure, we also note that for a small number of training data n, the error does not always decrease as the number of random features m increases. This indicates a delicate dependence of m as a function of n, in particular, n must increase with m; we observe the desired monotonic decrease in error with m when n is increased to 100 or 1000. In the over-parametrized regime, the authors in [53] presented a loose bound for this dependence for real-valued outputs. We leave a detailed account of the dependence of m on n required to achieve a certain error tolerance to future work.

Finally, Figure 6 demonstrates the invariance of the expected relative test error to the mesh resolution used for training and testing. This result is a consequence of framing the RFM on function space; other methods defined in finite-dimensions exhibit an *increase* in test error as mesh resolution is increased (see [11], Sec. 4, for a numerical account of this phenomenon). The first panel, Figure 6a, shows the error as a function of mesh resolution for three values of m. For very low resolution, the error varies slightly but then flattens out to a constant value as $K \to \infty$. More interestingly, these constant values of error, $e_{n',m} = 0.063$, 0.043, and 0.031 corresponding to $m = 256$, 512, and 1024, respectively, closely match the Monte Carlo rate



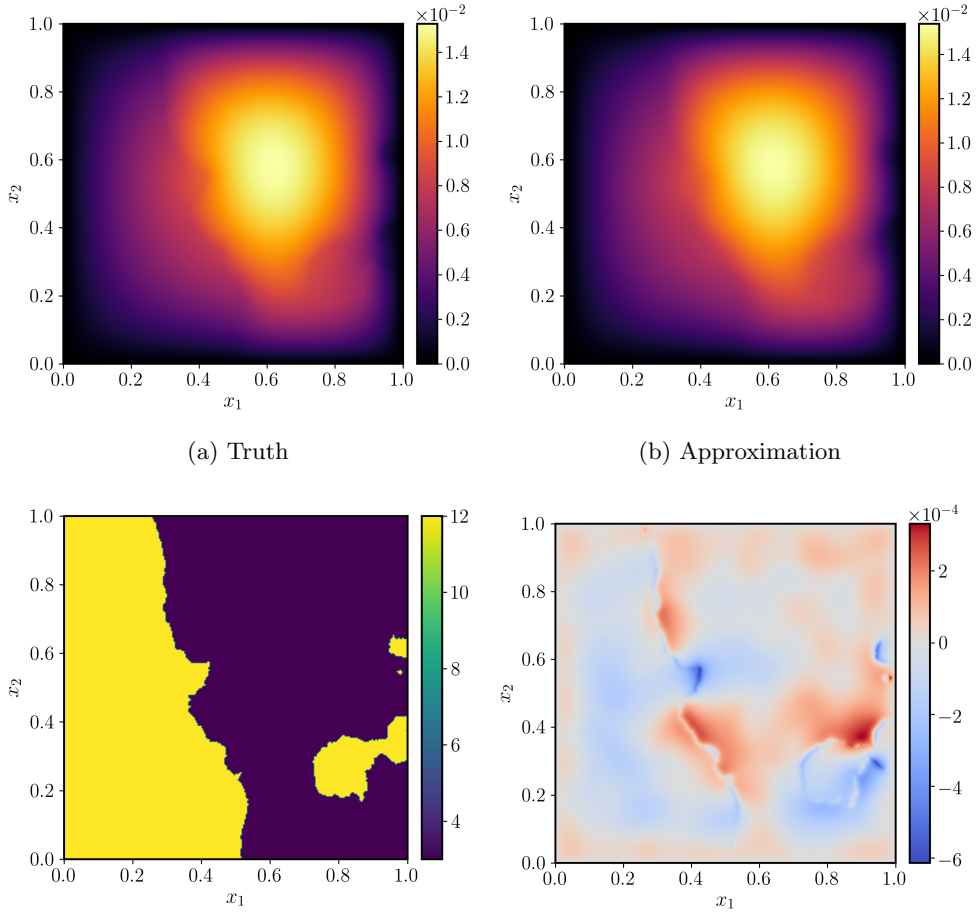
(a) (b)

Fig. 6: Results of a trained RFM for the Burgers' equation evolution operator $F^\dagger = \Psi_1$: Here, $n = 512$ training and $n' = 4000$ testing pairs were used. Figure 6a demonstrates resolution-invariant test error for various m; the error follows the $O(m^{-1/2})$ Monte Carlo rate remarkably well. Figure 6b displays the relative error of the learned coefficient α w.r.t. the coefficient learned on the highest mesh size ($K = 1025$).

$\mathcal{O}(m^{-1/2})$. While more theory is required to understand this behavior, it indicates that the optimization process is finding coefficients close to those arising from a Monte Carlo approximation of the ground truth map $F^\dagger$ as discussed in Subsection 2.3. The second panel, Figure 6b, indicates that the learned coefficient $\alpha^{(K)}$ for each K converges to some $\alpha^{(\infty)}$ as $K \to \infty$, again reflecting the design of the RFM as a mapping between infinite-dimensional spaces.

4.2. Darcy Flow: Experiment. In this section, we consider Darcy flow on the physical domain $D := (0,1)^2$, the unit square. We generate a high resolution dataset of input-output pairs by solving Equation (3.8) on a uniform 257×257 mesh (size $K = 257^2$) using a second order finite difference scheme. All mesh sizes $K < 257^2$ are subsampled from this original dataset and hence we consider numerical realizations of $F^\dagger$ up to $\mathbb{R}^{66049} \to \mathbb{R}^{66049}$. We denote *resolution* by r such that $K = r^2$. We fix $n = 128$ training and $n' = 1000$ testing pairs unless otherwise noted. The input data are drawn from the level set measure ν (3.9) with $\tau = 3$ and $\alpha = 2$ fixed. We choose $a^+ = 12$ and $a^- = 3$ in all experiments that follow and hence the contrast ratio $a^+/a^- = 4$ is fixed. The source is fixed to $f \equiv 1$, the constant function. We evaluate the predictor-corrector random features φ (3.12) using an FFT-based fast Poisson solver corresponding to an underlying second order finite difference stencil at a cost of $\mathcal{O}(K \log K)$ per solve. The smoothed coefficient a_ε in the definition of φ is obtained by solving (3.14) with time step 0.03 and diffusion constant $\eta = 10^{-4}$; with centered second order finite differences, this incurs 34 time steps and hence a cost $\mathcal{O}(34K)$. We fix the hyperparameters $\alpha' = 2$, $\tau' = 7.5$, $s^+ = 1/12$, $s^- = -1/3$, and $\delta = 0.15$ for the map φ. Unlike in Subsection 4.1, we find that regularization during training does improve the reconstruction of the Darcy flow solution operator and hence we train with $\lambda := 10^{-8}$ fixed. We remark that none of these hyperparameters were

optimized, so the RFM performance has the capacity to improve beyond the results we demonstrate in this section.



(a) Truth (b) Approximation

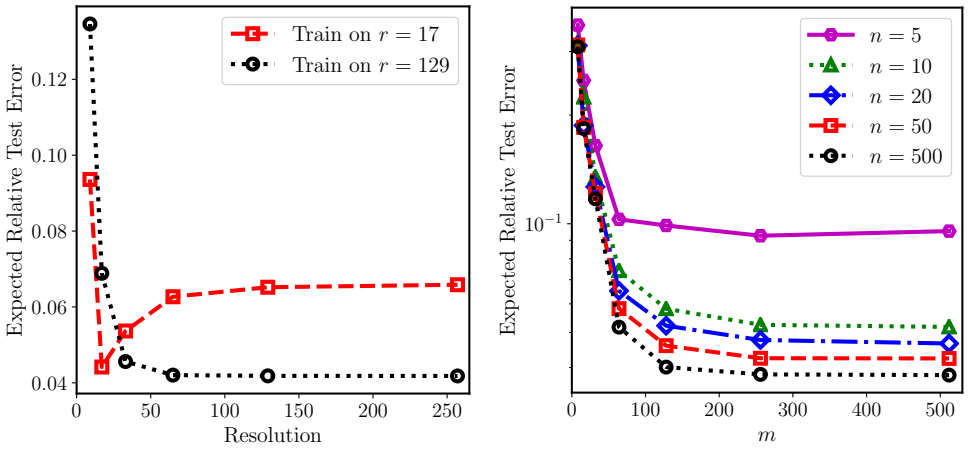
(c) Input (d) Pointwise Error

Fig. 7: Representative input-output test sample for the Darcy flow solution map: Here, $n = 256$, $m = 350$, and $K = 257^2$. Figure 7c shows a sample input, Figure 7a the resulting output (truth), Figure 7b a trained RFM prediction, and Figure 7d the pointwise error. The relative L^2 error for this single prediction is 0.0122.

Darcy flow is characterized by the geometry of the high contrast coefficients $a \sim \nu$. As seen in Figure 7, the solution inherits the steep interfaces of the input. However, we see that a trained RFM with predictor-corrector random features captures these interfaces well, albeit with slight smoothing; the error concentrates on the location of the interface. The effect of increasing m and n on the test error is shown in Figure 8b. Here, the error appears to saturate more than was observed for the Burgers' equation problem (Figure 5b). However, the smallest test error achieved for the best performing RFM configuration is 0.0381, which is competitive with competing approaches [11].

The RFM is able to successfully train and test on different resolutions for Darcy flow. Figure 8a shows that, again, for low resolutions, the smallest relative test

error is achieved when the train and test resolutions are identical (here, for $r = 17$); however, when the resolution is increased, the relative test error slightly increases then approaches a constant value, reflecting the function space design of the method. Training the RFM on a high resolution mesh poses no issues when transferring to lower resolution meshes for model evaluation, and it achieves consistent error for test resolutions sufficiently large ($r \geq 33$).

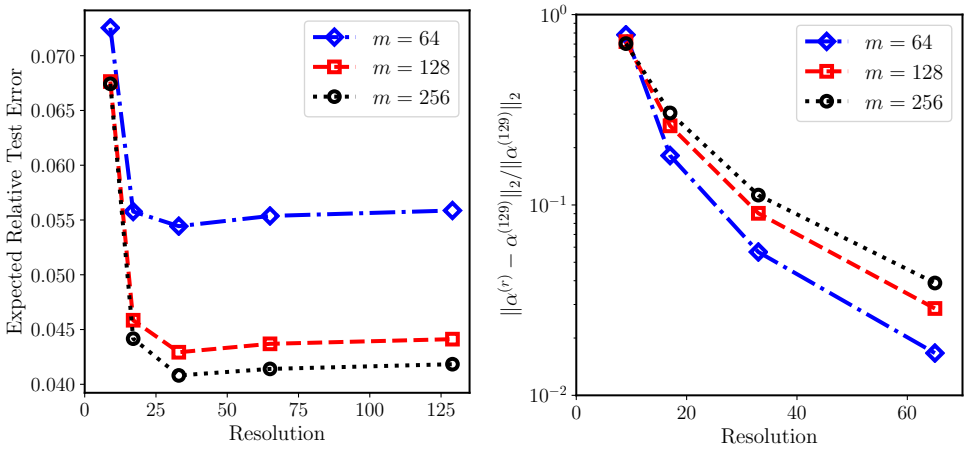


(a) (b)

Fig. 8: Expected relative test error of a trained RFM for Darcy flow with $n' = 1000$ test pairs: Figure 8a displays the invariance of test error w.r.t. training and testing on different resolutions for $m = 512$ and $n = 256$ fixed; the RFM can train and test on different mesh sizes without significant loss of accuracy. Figure 8b shows the decay of the test error for resolution $r = 33$ fixed as a function of m and n; the smallest error achieved is 0.0381 for $n = 500$ and $m = 512$.

In Figure 9, we again confirm that our method is invariant to the refinement of the mesh and improves with more random features. While the difference at low resolutions is more pronounced than that observed for Burgers' equation, our results for Darcy flow still suggest that the expected relative test error approaches a constant value as resolution increases; an estimate of this rate of convergence is seen in Figure 9b, where we plot the relative error of the learned parameter $\alpha^{(r)}$ at resolution r w.r.t. the parameter learned at the highest resolution trained, which was $r = 129$. Although we do not observe the limiting error following the Monte Carlo rate, which suggests that perhaps the RKHS $\mathcal{H}_{k_\mu}$ induced by the choice of φ is not expressive enough, the numerical results make clear that our methodology nonetheless performs well as a function approximator.

5. Conclusions. In this article, we introduced a random feature methodology for pure data-driven approximation of mappings $F^\dagger : \mathcal{X} \to \mathcal{Y}$ between infinite-dimensional Banach spaces. The random feature model $F_m(\cdot; \alpha^*)$, as an emulator of such maps, performs dimension reduction in the sense that the original problem of finding $F^\dagger$ is reduced to an approximate problem of finding m real numbers $\alpha^* \in \mathbb{R}^m$ (Section 2). While it does not immediately follow that the learned model $F_m(\cdot; \alpha^*)$ is necessarily cheaper to evaluate than a full order solver for $F^\dagger$, our design of problem-



(a) (b)

Fig. 9: Results of a trained RFM for Darcy flow: Here, $n = 128$ training and $n' = 1000$ testing pairs were used. Figure 9a demonstrates resolution-invariant test error for various m, while Figure 9b displays the relative error of the learned coefficient $\alpha^{(r)}$ at resolution r w.r.t. the coefficient learned on the highest resolution ($r = 129$).

specific random feature maps in Section 3 leads to efficient $\mathcal{O}(mK \log K)$ evaluation of the RFM for simple physical domain geometries and hence competitive computational cost in many-query settings.

Our non-intrusive methodology for high-dimensional approximation is one of only a select few [11, 48] that first designs a model in infinite dimensions, then discretizes; most other works in this line of research discretize the problem first, and then design a model between finite-dimensional spaces. Our conceptually infinite-dimensional algorithm results in a method that is consistent with the continuum picture, robust to discretization, and leads to more flexibility during practical use. For example, discretization of the input-output Banach spaces $\mathcal{X}$, $\mathcal{Y}$ is required for practical implementation and leads to high-dimensional functions $\mathbb{R}^K \to \mathbb{R}^K$. But as a method conceptualized on function space, the RFM is defined without any reference to a discretization and thus its approximation quality is consistent across different choices of mesh size K; indeed, the RFM could be trained on one method, say a spectral discretization, and deployed using another, say finite element or finite difference discretization. Furthermore, the RFM basis functions, that is, the random feature maps φ, are defined independently of the training data unlike in competing approaches such as the reduced basis method or the method in [11]; hence, our model may be directly evaluated on any mesh resolution once trained. These benefits were verified in numerical experiments for two nonlinear problems based on PDEs, one involving a semigroup and another a coefficient-to-solution operator (Section 4).

We remark that while our FFT implementations of the random feature maps in Section 3 have time complexity $\mathcal{O}(K \log K)$, this may be improved to the optimal linear $\mathcal{O}(K)$ with fast multipole or multigrid methods [31]. Although the method is implemented on uniform grids in space for speed and simplicity, the theoretical formulation we introduced for the RFM on function space holds irrespective of the discretization and hence an interesting extension of this work would design cheap-to-

evaluate random feature maps on unstructured grids, perhaps making the RFM more applicable to real experimental data or in applications outside of PDEs.

There are various other interesting directions for future work based on our random feature methodology. We are interested in application of the method to more challenging problems in the sciences such as climate modeling and material modeling, and to the solution of design and inverse problems arising in those settings, with the RFM serving as a cheap emulator. Furthermore, we wish to investigate a non-parametric generalization of the RFM inspired by moving least squares (MLS) [72]; MLS shares many parallels to the RFM and the reduced basis method that have yet to be explored. Also of interest is the question of allowing θ in $\varphi(\cdot;\theta)$ to adapt to data, for example, via sparsity constraints in the training of the RFM or solving a non-convex optimization problem obtained by choosing φ to be a neural network (designed in infinite dimensions) with trainable hidden parameters. Such explorations would serve to further clarify the effectiveness of function space learning algorithms. Finally, the development of a theory which underpins our method, and allows for proof of convergence, would be both mathematically challenging and highly desirable.

Acknowledgments. The authors are grateful to Bamdad Hosseini and Nikola B. Kovachki for helpful input which improved this work.

REFERENCES

[1] N. Aronszajn, *Theory of reproducing kernels*, Transactions of the American Mathematical Society, 68 (1950), pp. 337–404.

[2] F. Bach, *On the equivalence between kernel quadrature rules and random feature expansions*, The Journal of Machine Learning Research, 18 (2017), pp. 714–751.

[3] Y. Bar-Sinai, S. Hoyer, J. Hickey, and M. P. Brenner, *Learning data-driven discretizations for partial differential equations*, Proceedings of the National Academy of Sciences, 116 (2019), pp. 15344–15349.

[4] M. Barrault, Y. Maday, N. C. Nguyen, and A. T. Patera, *An empirical interpolation-method: application to efficient reduced-basis discretization of partial differential equations*, Comptes Rendus Mathematique, 339 (2004), pp. 667–672.

[5] J. Bear and M. Y. Corapcioglu, *Fundamentals of transport phenomena in porous media*, vol. 82, Springer Science & Business Media, 2012.

[6] M. Belkin, D. Hsu, S. Ma, and S. Mandal, *Reconciling modern machine-learning practice and the classical bias–variance trade-off*, Proceedings of the National Academy of Sciences, 116 (2019), pp. 15849–15854.

[7] P. Benner, A. Cohen, M. Ohlberger, and K. Willcox, *Model reduction and approximation: theory and algorithms*, vol. 15, SIAM, 2017.

[8] A. Berlinet and C. Thomas-Agnan, *Reproducing kernel Hilbert spaces in probability and statistics*, Springer Science & Business Media, 2011.

[9] C. Bernardi and R. Verfürth, *Adaptive finite element methods for elliptic equations with non-smooth coefficients*, Numerische Mathematik, 85 (2000), pp. 579–608.

[10] G. Beylkin and M. J. Mohlenkamp, *Algorithms for numerical analysis in high dimensions*, SIAM Journal on Scientific Computing, 26 (2005), pp. 2133–2159.

[11] K. Bhattacharya, B. Hosseini, N. B. Kovachki, and A. M. Stuart, *Model reduction and neural networks for parametric pdes*, arXiv preprint arXiv:2005.03180, (2020).

[12] D. Bigoni, Y. Chen, N. G. Trillos, Y. Marzouk, and D. Sanz-Alonso, *Data-driven forward discretizations for Bayesian inversion*, arXiv preprint arXiv:2003.07991, (2020).

[13] Y. Cao and Q. Gu, *Generalization bounds of stochastic gradient descent for wide and deep neural networks*, in Advances in Neural Information Processing Systems, 2019, pp. 10835–10845.

[14] C. Carmeli, E. De Vito, and A. Toigo, *Vector valued reproducing kernel Hilbert spaces of integrable functions and Mercer theorem*, Analysis and Applications, 4 (2006), pp. 377–408.

[15] G. Chen and K. Fidkowski, *Output-based error estimation and mesh adaptation using convolutional neural networks: Application to a scalar advection-diffusion problem*, in AIAA Scitech 2020 Forum, 2020, p. 1143.

[16] T. Chen and H. Chen, *Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems*, IEEE Transactions on Neural Networks, 6 (1995), pp. 911–917.
[17] M. Cheng, T. Y. Hou, M. Yan, and Z. Zhang, *A data-driven stochastic method for elliptic PDEs with random coefficients*, SIAM/ASA Journal on Uncertainty Quantification, 1 (2013), pp. 452–493.
[18] A. Chkifa, A. Cohen, R. DeVore, and C. Schwab, *Sparse adaptive taylor approximation algorithms for parametric and stochastic elliptic pdes*, ESAIM: Mathematical Modelling and Numerical Analysis, 47 (2013), pp. 253–280.
[19] A. Cohen and R. DeVore, *Approximation of high-dimensional parametric PDEs*, Acta Numerica, 24 (2015), pp. 1–159.
[20] A. Cohen and G. Migliorati, *Optimal weighted least-squares methods*, arXiv preprint arXiv:1608.00512, (2016).
[21] S. L. Cotter, G. O. Roberts, A. M. Stuart, and D. White, *Mcmc methods for functions: modifying old algorithms to make them faster*, Statistical Science, (2013), pp. 424–446.
[22] F. Cucker and S. Smale, *On the mathematical foundations of learning*, Bulletin of the American mathematical society, 39 (2002), pp. 1–49.
[23] L. Demanet, *Curvelets, wave atoms, and wave equations*, PhD thesis, California Institute of Technology, 2006.
[24] R. A. DeVore, *The theoretical foundation of reduced basis methods*, Model Reduction and approximation: Theory and Algorithms, (2014), pp. 137–168.
[25] A. Doostan and G. Iaccarino, *A least-squares approximation of partial differential equations with high-dimensional random inputs*, Journal of Computational Physics, 228 (2009), pp. 4332–4345.
[26] L. C. Evans, *Partial differential equations*, vol. 19, American Mathematical Soc., 2010.
[27] Y. Fan and L. Ying, *Solving electrical impedance tomography with deep learning*, Journal of Computational Physics, 404 (2020), pp. 109–119.
[28] J. Feliu-Faba, Y. Fan, and L. Ying, *Meta-learning pseudo-differential operators with deep neural networks*, Journal of Computational Physics, 408 (2020), p. 109309.
[29] H. Gao, J.-X. Wang, and M. J. Zahr, *Non-intrusive model reduction of large-scale, nonlinear dynamical systems using deep learning*, arXiv preprint arXiv:1911.03808, (2019).
[30] M. Geist, P. Petersen, M. Raslan, R. Schneider, and G. Kutyniok, *Numerical solution of the parametric diffusion equation by deep neural networks*, arXiv preprint arXiv:2004.12131, (2020).
[31] A. Gholami, D. Malhotra, H. Sundar, and G. Biros, *Fft, fmm, or multigrid? a comparative study of state-of-the-art poisson solvers for uniform and nonuniform grids in the unit cube*, SIAM Journal on Scientific Computing, 38 (2016), pp. C280–C306.
[32] D. Gilbarg and N. S. Trudinger, *Elliptic partial differential equations of second order*, springer, 2015.
[33] R. Gonzalez-Garcia, R. Rico-Martinez, and I. Kevrekidis, *Identification of distributed parameter systems: A neural net based approach*, Computers & chemical engineering, 22 (1998), pp. S965–S968.
[34] M. Griebel and C. Rieger, *Reproducing kernel Hilbert spaces for parametric partial differential equations*, SIAM/ASA Journal on Uncertainty Quantification, 5 (2017), pp. 111–137.
[35] E. Haber and L. Ruthotto, *Stable architectures for deep neural networks*, Inverse Problems, 34 (2017), p. 014004.
[36] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: data mining, inference, and prediction*, Springer Science & Business Media, 2009.
[37] J. S. Hesthaven and S. Ubbiali, *Non-intrusive reduced order modeling of nonlinear problems using neural networks*, Journal of Computational Physics, 363 (2018), pp. 55–78.
[38] M. Hinze, R. Pinnau, M. Ulbrich, and S. Ulbrich, *Optimization with PDE constraints*, vol. 23, Springer Science & Business Media, 2008.
[39] A. Jacot, F. Gabriel, and C. Hongler, *Neural tangent kernel: Convergence and generalization in neural networks*, in Advances in neural information processing systems, 2018, pp. 8571–8580.
[40] H. Kadri, E. Duflos, P. Preux, S. Canu, A. Rakotomamonjy, and J. Audiffren, *Operator-valued kernels for learning from functional response data*, The Journal of Machine Learning Research, 17 (2016), pp. 613–666.
[41] A.-K. Kassam and L. N. Trefethen, *Fourth-order time-stepping for stiff PDEs*, SIAM Journal on Scientific Computing, 26 (2005), pp. 1214–1233.
[42] R. Kempf, H. Wendland, and C. Rieger, *Kernel-based reconstructions for parametric PDEs*, in International Workshop on Meshfree Methods for Partial Differential Equations,

Springer, 2017, pp. 53–71.
[43] Y. Khoo, J. Lu, and L. Ying, *Solving parametric pde problems with artificial neural networks*, arXiv preprint arXiv:1707.03351, (2017).
[44] A. Kiselev, F. Nazarov, and R. Shterenberg, *Blow up and regularity for fractal burgers equation*, arXiv preprint arXiv:0804.3549, (2008).
[45] G. Kutyniok, P. Petersen, M. Raslan, and R. Schneider, *A theoretical analysis of deep neural networks and parametric PDEs*, arXiv preprint arXiv:1904.00377, (2019).
[46] K. Lee and K. T. Carlberg, *Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders*, Journal of Computational Physics, 404 (2020), p. 108973.
[47] Y. Li, J. Lu, and A. Mao, *Variational training of neural network approximations of solution maps for physical models*, Journal of Computational Physics, 409 (2020), p. 109338.
[48] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar, *Neural operator: Graph kernel network for partial differential equations*, arXiv preprint arXiv:2003.03485, (2020).
[49] Z. Long, Y. Lu, X. Ma, and B. Dong, *Pde-net: Learning PDEs from data*, arXiv preprint arXiv:1710.09668, (2017).
[50] L. Lu, P. Jin, and G. E. Karniadakis, *Deeponet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators*, arXiv preprint arXiv:1910.03193, (2019).
[51] D. G. Luenberger, *Optimization by vector space methods*, John Wiley & Sons, 1997.
[52] C. Ma, L. Wu, and E. Weinan, *Machine learning from a continuous viewpoint*, arXiv preprint arXiv:1912.12777, (2019).
[53] C. Ma, L. Wu, and E. Weinan, *On the generalization properties of minimum-norm solutions for over-parameterized neural network models*, arXiv preprint arXiv:1912.06987, (2019).
[54] C. A. Micchelli and M. Pontil, *On learning vector-valued functions*, Neural computation, 17 (2005), pp. 177–204.
[55] R. G. Patel and O. Desjardins, *Nonlinear integro-differential operator regression with neural networks*, arXiv preprint arXiv:1810.08552, (2018).
[56] B. Peherstorfer, K. Willcox, and M. Gunzburger, *Survey of multifidelity methods in uncertainty propagation, inference, and optimization*, Siam Review, 60 (2018), pp. 550–591.
[57] A. Rahimi and B. Recht, *Random features for large-scale kernel machines*, in Advances in neural information processing systems, 2008, pp. 1177–1184.
[58] M. Raissi, P. Perdikaris, and G. E. Karniadakis, *Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations*, Journal of Computational Physics, 378 (2019), pp. 686–707.
[59] R. Rico-Martinez, K. Krischer, I. Kevrekidis, M. Kube, and J. Hudson, *Discrete-vs. continuous-time nonlinear signal processing of cu electrodissolution data*, Chemical Engineering Communications, 118 (1992), pp. 25–48.
[60] F. Rossi and B. Conan-Guez, *Functional multi-layer perceptron: a non-linear tool for functional data analysis*, Neural networks, 18 (2005), pp. 45–60.
[61] L. Ruthotto and E. Haber, *Deep neural networks motivated by partial differential equations*, Journal of Mathematical Imaging and Vision, (2019), pp. 1–13.
[62] N. D. Santo, S. Deparis, and L. Pegolotti, *Data driven approximation of parametrized PDEs by reduced basis and neural networks*, arXiv preprint arXiv:1904.01514, (2019).
[63] C. Schwab and J. Zech, *Deep learning in high dimension: Neural network expression rates for generalized polynomial chaos expansions in uq*, Analysis and Applications, 17 (2019), pp. 19–55.
[64] J. Sirignano and K. Spiliopoulos, *Dgm: A deep learning algorithm for solving partial differential equations*, Journal of Computational Physics, 375 (2018), pp. 1339–1364.
[65] P. D. Spanos and R. Ghanem, *Stochastic finite element expansion for random media*, Journal of engineering mechanics, 115 (1989), pp. 1035–1053.
[66] Y. Sun, A. Gilbert, and A. Tewari, *Random relu features: Universality, approximation, and composition*, stat, 1050 (2018), p. 10.
[67] N. Trask, R. G. Patel, B. J. Gross, and P. J. Atzberger, *Gmls-nets: A framework for learning from unstructured data*, arXiv preprint arXiv:1909.05371, (2019).
[68] R. K. Tripathy and I. Bilionis, *Deep uq: Learning deep neural network surrogate models for high dimensional uncertainty quantification*, Journal of computational physics, 375 (2018), pp. 565–588.
[69] E. Weinan, *A proposal on machine learning via dynamical systems*, Communications in Mathematics and Statistics, 5 (2017), pp. 1–11.

[70] E. Weinan, J. Han, and Q. Li, *A mean-field optimal control formulation of deep learning*, Research in the Mathematical Sciences, 6 (2019), p. 10.
[71] E. Weinan and B. Yu, *The deep ritz method: a deep learning-based numerical algorithm for solving variational problems*, Communications in Mathematics and Statistics, 6 (2018), pp. 1–12.
[72] H. Wendland, *Scattered data approximation*, vol. 17, Cambridge university press, 2004.
[73] C. K. Williams, *Computing with infinite networks*, in Advances in neural information processing systems, 1997, pp. 295–301.
[74] C. K. Williams and C. E. Rasmussen, *Gaussian processes for machine learning*, vol. 2, MIT press Cambridge, MA, 2006.
[75] N. Winovich, K. Ramani, and G. Lin, *Convpde-uq: Convolutional neural networks with quantified uncertainty for heterogeneous elliptic partial differential equations on varied domains*, Journal of Computational Physics, 394 (2019), pp. 263–279.
[76] Y. Zhu and N. Zabaras, *Bayesian deep convolutional encoder–decoder networks for surrogate modeling and uncertainty quantification*, Journal of Computational Physics, 366 (2018), pp. 415–447.

Appendix A. Proofs of Results.

Proof of Result 2.4. Fix $a \in \mathcal{X}$ and $y \in \mathcal{Y}$. Then, we note that

$$k_\mu(\cdot, a)y = \int \langle \varphi(a;\theta), y\rangle_{\mathcal{Y}} \varphi(\cdot;\theta)\mu(d\theta) = \mathcal{A}\langle \varphi(a;\cdot), y\rangle_{\mathcal{Y}} \in \mathrm{im}(\mathcal{A})\,, \tag{A.1}$$

since $\langle \varphi(a;\cdot), y\rangle_{\mathcal{Y}} \in L^2_\mu(\Theta;\mathbb{R})$ by the Cauchy-Schwarz inequality.

Now, we show that $\mathrm{im}(\mathcal{A})$ admits a reproducing property of the form (2.9). First, note that $\mathcal{A}$ can be viewed as a bijection between its coimage and image spaces, and we denote this bijection by $\tilde{\mathcal{A}}$:

$$\tilde{\mathcal{A}} : \ker(\mathcal{A})^\perp \to \mathrm{im}(\mathcal{A})\,. \tag{A.2}$$

For any $F, G \in \mathrm{im}(\mathcal{A})$, define the candidate RKHS inner product $\langle \cdot, \cdot \rangle$ by

$$\langle F, G\rangle := \langle \tilde{\mathcal{A}}^{-1}F, \tilde{\mathcal{A}}^{-1}G\rangle_{L^2_\mu(\Theta;\mathbb{R})}\,; \tag{A.3}$$

this is indeed a valid inner product since $\tilde{\mathcal{A}}$ is invertible. Note that for any $q \in \ker(\mathcal{A})$,

$$\begin{aligned}
\left\langle q, \langle \varphi(a;\cdot), y\rangle_{\mathcal{Y}} \right\rangle_{L^2_\mu(\Theta;\mathbb{R})} &= \int q(\theta)\langle \varphi(a;\theta), y\rangle_{\mathcal{Y}}\, \mu(d\theta) \\
&= \left\langle \int q(\theta)\varphi(a;\theta)\mu(d\theta), y \right\rangle_{\mathcal{Y}} \\
&= 0
\end{aligned}$$

so that $\langle \varphi(a;\cdot), y\rangle_{\mathcal{Y}} \in \ker(\mathcal{A})^\perp$. Then, we compute

$$\begin{aligned}
\langle k_\mu(\cdot, a)y, F\rangle &= \left\langle \langle \varphi(a;\cdot), y\rangle_{\mathcal{Y}}, \tilde{\mathcal{A}}^{-1}F \right\rangle_{L^2_\mu(\Theta;\mathbb{R})} \\
&= \int \langle \varphi(a;\theta), y\rangle_{\mathcal{Y}} (\tilde{\mathcal{A}}^{-1}F)(\theta)\mu(d\theta) \\
&= \left\langle \int (\tilde{\mathcal{A}}^{-1}F)(\theta)\varphi(a;\theta)\mu(d\theta), y \right\rangle_{\mathcal{Y}} \\
&= \left\langle y, (\mathcal{A}\tilde{\mathcal{A}}^{-1}F)(a) \right\rangle_{\mathcal{Y}} \\
&= \langle y, F(a)\rangle_{\mathcal{Y}}\,,
\end{aligned}$$

which gives exactly (2.9) if our candidate inner product is defined to be the RKHS inner product. Since $F \in \mathrm{im}(\mathcal{A})$ is arbitrary, this and (A.1) together imply that $\mathrm{im}(\mathcal{A}) = H_{k_\mu}$ is the RKHS induced by k_μ as shown in [22, 40]. ∎

Proof of Result 2.5. Since $L^2_{\mu^{(m)}}(\Theta;\mathbb{R})$ is isomorphic to $\mathbb{R}^m$, we can consider the map $\mathcal{A}:\mathbb{R}^m \to L^2_\nu(\mathcal{X};\mathcal{Y})$ defined in (2.13) and use Result 2.4 to conclude that

$$\mathcal{H}_{k^{(m)}} = \mathrm{im}(\mathcal{A}) = \left\{ \frac{1}{m}\sum_{j=1}^m c_j \varphi(\cdot;\theta_j) : c \in \mathbb{R}^m \right\} = \mathrm{span}\{\varphi_j\}_{j=1}^m , \tag{A.4}$$

since the $\{\varphi_j\}_{j=1}^m$ are assumed linearly independent. ∎

Proof of Result 2.7. Recall from Result 2.5 that the RKHS $\mathcal{H}_{k^{(m)}}$ comprises the linear span of the $\{\varphi_j := \varphi(\cdot;\theta_j)\}_{j=1}^m$. Hence $\varphi_j \in \mathcal{H}_{k^{(m)}}$, and note that by the reproducing kernel property (2.9), for any $F \in \mathcal{H}_{k^{(m)}}$, $a \in \mathcal{X}$ and $y \in \mathcal{Y}$,

$$\begin{aligned}
\langle y, F(a)\rangle_{\mathcal{Y}} &= \left\langle k^{(m)}(\cdot,a)y, F \right\rangle_{\mathcal{H}_{k^{(m)}}} \\
&= \frac{1}{m}\sum_{j=1}^m \langle \varphi_j(a), y\rangle_{\mathcal{Y}} \langle \varphi_j, F\rangle_{\mathcal{H}_{k^{(m)}}} \\
&= \left\langle y, \frac{1}{m}\sum_{j=1}^m \langle \varphi_j, F\rangle_{\mathcal{H}_{k^{(m)}}} \varphi_j(a) \right\rangle.
\end{aligned}$$

Since this is true for all $y \in \mathcal{Y}$, we deduce that

$$F = \frac{1}{m}\sum_{j=1}^m \alpha_j \varphi_j , \tag{A.5a}$$

$$\alpha_j = \langle \varphi_j, F\rangle_{\mathcal{H}_{k^{(m)}}} . \tag{A.5b}$$

Since $\{\varphi_j\}_{j=1}^m$ are assumed linearly independent, we deduce that the representation (A.5) is unique.

Finally, we calculate the RKHS norm of any such F in terms of α. Note

$$\begin{aligned}
\|F\|^2_{\mathcal{H}_{k^{(m)}}} = \langle F, F\rangle_{\mathcal{H}_{k^{(m)}}} &= \left\langle \frac{1}{m}\sum_{j=1}^m \alpha_j \varphi_j, F \right\rangle_{\mathcal{H}_{k^{(m)}}} \\
&= \frac{1}{m}\sum_{j=1}^m \alpha_j \langle \varphi_j, F\rangle_{\mathcal{H}_{k^{(m)}}} \\
&= \frac{1}{m}\sum_{j=1}^m \alpha_j^2 .
\end{aligned}$$

Substituting this into (2.23), we obtain the desired equivalence with (2.22). ∎